

DOCUMENT RESUME

ED 304 590

CE 052 134

AUTHOR Mills, Steven; And Others
TITLE Welcome to dBase III Plus Advanced. Learning Activity Packets.
INSTITUTION Oklahoma State Dept. of Vocational and Technical Education, Stillwater. Curriculum and Instructional Materials Center.
PUB DATE 89
NOTE 108p.; For a related document, see CE 052 133.
AVAILABLE FROM Curriculum and Instructional Materials Center, Oklahoma Department of Vocational and Technical Education, 1500 West Seventh Avenue, Stillwater, OK 74074-4364 (Order No. BO-3021).
PUB TYPE Guides - Classroom Use - Guides (For Teachers) (052)
EDRS PRICE MF01/PC05 Plus Postage.
DESCRIPTORS *Business Education; Computer Assisted Instruction; Computer Software; Database Design; *Database Management Systems; *Databases; Data Processing; *Individualized Instruction; Learning Activities; Learning Modules; Management Systems; Office Occupations Education; Pacing; Postsecondary Education; Secondary Education
IDENTIFIERS *dBase III Plus

ABSTRACT

This learning activity packet (LAP) contains five self-paced study lessons that allow students to proceed along a 22-hour course of study for dBase III Plus Advanced at their own pace. It continues the introductory dBase III Plus course. The lessons are organized in the following way: objectives, completion standard, performance standard, a list of required materials, unit test, and exercises (applications of the operations to measure the laboratory competency of the student in regard to the materials presented in the LAP). Most lessons are presented in a "read and do" format. The lessons are: linking and updating multiple files; inputting and outputting to other programs; writing simple programs; system design and program development; and applications generator. Unit tests for each lesson and answer keys for the tests and all exercises are appended. These lessons have been prepared for use with the IBM Personal Computer. (YLB)

* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *

LEARNING ACTIVITY PACKETS

U.S. DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION
CENTER (ERIC)

☒ This document has been reproduced as
received from the person or organization
originating it.

☐ Minor changes have been made to improve
reproduction quality.

• Points of view or opinions stated in this docu-
ment do not necessarily represent official
OERI position or policy.

"PERMISSION TO REPRODUCE THIS
MATERIAL HAS BEEN GRANTED BY

R. Malin

TO THE EDUCATIONAL RESOURCES
INFORMATION CENTER (ERIC)."



Developed by
Curriculum and Instructional Materials Center
Oklahoma State Department of Vocational and Technical Education
Dr. Roy Peters, Jr., Director
Oklahoma State Board of Vocational and Technical Education
Gerald E. Hoeltzel, Chairman

LEARNING ACTIVITY PACKETS



Developed by
Curriculum and Instructional Materials Center
Oklahoma State Department of Vocational and Technical Education
Dr. Roy Peters, Jr., Director
Oklahoma State Board of Vocational and Technical Education
Gerald E. Hocltzel, Chairman

ACKNOWLEDGEMENTS

The development of the *dBase III Plus*® *Advanced Learning Activities Packet* is credited to Steven Mills, Eddie Palovik and Gayle Kearns. It was their hard work and effort that made the LAPs a success.

A great appreciation is extended to Greg Pierce, CIMC Coordinator and Sheila Stone, Curriculum Development Coordinator for their coordination during the development of the Learning Activity Packets. A special thanks goes to Terrell Ogilvie, Technical Writer, for his contribution toward the development of the LAPs.

A special thanks goes to the Business and Office instructors who participated in the training workshops and helped validate and refine the LAPs.

PREFACE

WHAT ARE LAPS?

Learning Activity Packets (LAPs) are self-paced study lessons that allow the student to proceed along a course of study at his or her own pace. The student is allowed to progress to the next lesson in the unit only after a certain level of competency and understanding is achieved.

The advantage to the student of using LAPs for certain areas of study is obvious in that it allows the student to progress at his or her own pace. Some students learn faster than others and LAPs takes this fact into consideration by allowing all to learn at their own rate.

The advantage to the instructor of using LAPs for certain areas of study is that the instructor becomes available to be your lab partner. This allows more individual attention to each student's learning needs.

ORGANIZATION OF LESSONS

The lessons used in the LAPs are organized in the following way:

LESSON NAME	This is the name of the LAP within the instructional unit.
LESSON NO.	Each lesson is numbered to maintain the right sequence of instruction
OBJECTIVES	A description of the major points or concepts which will be taught in a lesson as well as a description of what the student should be able to perform upon completion of the unit.
COMPLETION STANDARD	This is a suggested maximum amount of time the student will need in order to complete the lesson. This information is available in the Teachers Manual and Index.
PERFORMANCE STANDARD	This score must be achieved on the Unit Test and Exercises before progressing to the next unit. These LAPs will always use a standard of 70% for each lesson.
REQUIRED MATERIALS	This is a list of all materials the student will need in order to complete the lesson.

UNIT TEST	The Unit Test is used to measure the theory competency of the student in regard to the material presented in the LAP.
EXERCISES	An application of the operations is assigned to measure the laboratory competency of the student in regard to the material presented in the LAP.

READ AND DO

Most lessons are presented in a READ and DO format. Some information is presented in a READ paragraph. A response to the information just presented is made in the form of a DO paragraph. However, sometimes information is presented with no requisite DO responses.

COMPUTER HARDWARE CONFIGURATION

These lessons have been prepared for use with the IBM Personal Computer. Since the computer you use may be equipped differently from other IBM-PC's, you should take note as to the configuration of your computer. Many instructions may refer to a particular disk configuration or a particular kind of printer. You should be aware of the particular configuration for the microcomputer you are using.

SOFTWARE VERSION

The information presented in the lessons generally reflect the latest version of the software currently available on the consumer market.

LEARNING PROCEDURES

This is the step-by-step procedure the student should follow in order to complete the lesson properly. Generally, the learning procedure for each lesson should follow this schedule:

1. READ the Objectives for the lesson. Be certain that you have selected the correct lesson and have the required materials.
2. READ and DO the information and operations presented in the lesson.
3. DO the Exercises at the end of the lesson and return to the instructor for scoring.
4. STUDY the information presented in the lesson.
5. DO the Unit Test and return to the instructor for scoring.
6. If you meet the performance standard, MOVE to the next lesson in the sequence.

dBASE III PLUS -- ADVANCED

NO.	LESSON NAME	PRE REQ.	COMPLETE. STANDARD
10	Linking and Updating Multiple Files	01-9	3 hrs.
11	Inputting and Outputting to Other Programs	10	5 hrs.
12	Writing Simple Programs	11	5 hrs.
13	System Design and Program Development	12	4 hrs.
14	Applications Generator	13	5 hrs.
TOTAL TRAINING TIME FOR COURSE			22 hrs.

LESSON 10

LINKING AND UPDATING MULTIPLE FILES

OBJECTIVES:

- * Selecting database file work areas
- * Linking multiple database files
- * Updating from one database file to another
- * Printing reports using linked databases
- * Clearing the database work areas

REQUIRED MATERIALS:

- * Paper and pen or pencil
- * dBASE III PLUS System Diskettes 1 and 2
- * A practice diskette
- * IBM PC/XT, PS/II, or compatible with 2 disk drives or 1 disk drive and hard disk, and printer

READ: SELECTING DATABASE FILE WORK AREAS

dBASE III PLUS allows the simultaneous use of up to 10 database files. Each of these can be used independently, as if it were the only file in use. While working on one file, the current record in any of the other files can be read, but not changed.

Separate buffers or work areas are provided for each database file to be used. These areas are identified as 1 through 10 or, alternately, A through J. When dBASE III PLUS is entered, the work area is 1. To work with multiple database files, a separate work area must be identified as each database file is opened (USED).

Work areas are selected by means of the SELECT command. Work areas do not have to be selected in order. In the example below, the initial work area is work area "1." Students is identified as the file to be used in that work area. Work area "2" is then selected as the active work area. Teachers is identified as the file to be used in this work area. For example:

```
.USE Students (Students use work area 1.)
.SELECT 2
.USE Teachers (Teachers use work area 2.)
```

At this time, work area 2 is the active work area. The active database file is Teachers. All normal database operations can be performed within that file. Most importantly, records can be selected and altered. The current record in Students can be accessed (read only). Record movement is permitted in only the active file.

Once a file is opened in a work area, the work area may be selected by either the work area number (as shown above) or by use of the assigned file Alias. To illustrate, the file Students in work area 1 may be selected by:

```
.SELECT Students or .SELECT 1 or .SELECT A
```

READ: LINKING MULTIPLE DATABASE FILES

Two files may be linked on the basis of a common field or expression. The second file must be indexed on the common field or expression.

For example, the two files below, Students and Teachers, each contain the fields Name, Grade, and Room:

STUDENTS

REC.NO.	ROOM	GRADE	STUDENT
0001	11	2	Adams, S.
0002	15	1	Allen, B.
0003	9	3	Brown, D.
0004	7		
0005	14		
0006	18		
0007	12		

TEACHERS

REC.NO.	ROOM	GRADE	TEACHER
0001	14	4	Abernathy, A.
0002	11	2	Baker, W.
0003	7	5	Carson, J.

If the file Teachers is indexed by Room, Teachers may be linked to the active file Students by:

```
.SET RELATION TO Room INTO Teachers
```

Whenever the file Students is repositioned, dBASE III PLUS will reposition the Teachers file to a record that matches the content of the Students Room.

If no matching record exists, Teachers is repositioned to a blank record. This allows two files to be used together in business applications and in preparing reports. For example, the Teacher's Name can be displayed along with the Student's Name, Room, and Grade by:

.DISPLAY Student, Room, Grade, Teachers->Teacher

To include a field that is not in the active database, it must be identified as **alias->field**.

READ: UPDATING FROM ONE DATABASE FILE TO ANOTHER

Selecting multiple files is quite useful when updating the information from one database file to another. The UPDATE command uses data from an identified source file to make changes to data records in the active database file, the target file. The changes are accomplished by matching records in the two database files on a single key field.

When using the UPDATE command, the database being UPDATED (the target database) must be the active database file in the currently selected work area. The source database file (the one doing the updating--the one being updated from) must be active in one of the unselected work areas. The key field must be the same (with the same field name) in both database files.

The illustration below depicts updating the target database file, Stock, from the source file, Invoice. The key field that links these two databases is Item. Both databases are sorted by Item. The contents of the source field, Qty, is to be added to the content of the target field, Qty, for matching items. Stock is the active database:

```
.SELECT 2
.USE Invoice
.SELECT 1
.USE Stock
.UPDATE ON Item FROM Invoice REPLACE Qty WITH
Qty+Invoice->Qty
```

STOCK		+	INVOICE		=	STOCK	
ITEM	QTY		ITEM	QTY		ITEM	QTY
Hammer	3		Hammer	2		Hammer	5
Pliers	2		Pulley	2		Pliers	2
Pulley	1		Saw	10		Pulley	3
Saw	10					Saw	20
Socket	5					Socket	5

When the REPLACE expression includes a field that is in the source database (which is usually the case), the source field must be identified as: alias->field.

READ: LINKING FILES AND PRINTING A REPORT

An application that would require the linking of two files would be a report that would list limousine information from the LIMO file and the corresponding customer name for each automobile rented from the CUST file. The common or key field that these two database files are linked on is CUST, the customer ID field.

When the report is printed, each record from LIMO, the active database file, will be printed, and the record pointer will be repositioned to the corresponding customer record (based on the customer ID field, CUST) in the CUST database file in order to retrieve the customer name for printing.

DO: SELECT A DATABASE FILE

Let's index the customer database, CUST. If dBASE III PLUS is not active, you should start the program, set the default drive, and set the editing menus on. In order to index a database file you must have an active database file:

1. Position the selection bar on Set Up.
2. Position the selection bar on Database file and press <Return>.
3. Position the selection bar on B: (or A: for hard disk) and press <Return>.
4. Position the selection bar on CUST.DBF and press <Return>.
5. At the prompt, Is the file indexed?, press <Return>.

INTERACTIVE MODE (from the dot prompt):

Type: .USE CUST <Return>

DO: CREATING AN INDEX FILE

In order to set the relationship CUST must be indexed by the control or key field, CUST. Then the database file is indexed by the customer id field, CUST, to an index file named CUSTNDX.

Finally, the database file is re-opened or USED with an active index.

1. Position the selection bar on **Organize**.
2. Position the selection bar on **Index** and press **<Return>**.
3. Press **<F10>** to display the Fields menu on the screen.
4. Position the selection bar on the field name **CUST** and press **<Return>** and then press **<Return>** again in response to the completed prompt.
5. Position the selection bar on **B:** (or **A:** for hard disk) and press **<Return>**.
6. At the prompt, **Enter the name of the file:**, type **CUSTNDX**.

Notice the command you have now built:

INDEX ON CUST TO B:(or A:)CUSTNDX

7. Now position the selection bar on **Set Up**.
8. Position the selection bar on **Database file** and press **<Return>**.
9. Position the selection bar on **B:** (or **A:** for hard disk) and press **<Return>**.
10. Position the selection bar on **CUST.DBF** and press **<Return>**.
11. At the prompt, **Is the file indexed?**, type **Y**.
12. Position the selection bar on **CUSTNDX.NDX** and press **<Return>**.

DO: SELECT THE WORK AREAS

Next, both databases must be assigned a work area with **LIMO** being the active database and **CUST** occupying an unselected work area.

Press **<Esc>** to leave **Assist** because you will not be able to use **Assist** to perform the following commands:

Type: **.SELECT 1 <Return>**

Type: **.USE LIMO <Return>**

Type: **.SELECT 2 <Return>**

Type: .USE CUST INDEX CUSTNDX <Return>

Type: .SELECT 1 <Return>

DO: SET THE RELATIONSHIP

Then, set the relationship based on the common field, CUST. LIMO is the active database so the relationship will be set into CUST in an unselected work area:

Type: .SET RELATION TO CUST INTO CUST <Return>

Now return to the Assist program:

Type: .ASSIST <Return>

DO: CREATE AND PRINT A REPORT USING LINKED DATABASES

Now create a report named LIMOCUST that will use information from both the LIMO and CUST databases:

1. Position the selection bar on Create.
2. Position the selection bar on Report and press <Return>.
3. Position the selection bar on B: (or A: for hard disk) and press <Return>.
4. At the prompt, Enter the name of the file:, type LIMOCUST.

INTERACTIVE MODE (from the dot prompt):

Type: .CREATE REPORT LIMOCUST <Return>

5. Create a report that prints the LICENSE, MAKE, YEAR, CUST, and a column with the last and first name from the CUST database file with the field name, TRIM(CUST->LAST)+", "+TRIM(CUST->FIRST).

Print the report only for those limousines which are rented--that is, those with a customer id not equal to zero:

6. Position the selection bar on Retrieve.
7. Position the selection bar on Report and press <Return>.

8. Position the selection bar on B: (or A: for hard disk) and press <Return>.
9. Position the selection bar on LIMOCUST.FRM and press <Return>.
10. Position the selection bar on Build a search condition and press <Return>.
11. Position the selection bar on CUST and press <Return>.
12. Position the selection bar on Greater than and press <Return>.
13. At the prompt, Enter a numeric value, type 0.
14. Position the selection bar on No more conditions and press <Return>.
15. Position the selection bar on Execute the command and press <Return>.
16. At the prompt, Direct the output to the printer? Y/N, type Y.

INTERACTIVE MODE (from the dot prompt):

Type: .REPORT FORM LIMOCUST FOR CUST>0 TO PRINT <Return>

DO: CLEAR THE WORK AREAS

Upon completing this Operation Sheet, you can clear all work area and close all databases with the following command.

Press <Esc> to leave Assist because you will not be able to use Assist to perform the following commands:

Type: .CLOSE DATABASES <Return>

Now return to the Assist program:

Type: .ASSIST <Return>

READ: UPDATING FROM ONE DATABASE FILE TO ANOTHER

Every month the customer file must be updated with the month's billings. You must take the amount in the RENTAL field in the LIMO database file and add it to the RENTAL field in the CUST

file. Those fields have the same name in the two files to facilitate updating the customer file. In the LIMO file, RENTAL contains the monthly charge to be levied the customer leasing that car. In the CUST file, RENTAL contains the accumulated monthly billings since the customer first rented the car.

In order to perform the update operation, both files must be in the same order, so start by sorting CUST to CUSTSORT and LIMO to LIMOSORT. The control field will be CUST in both cases. These sorting operations will produce two files, both in ascending sequence by the contents of the field CUST.

DO: SORT THE DATABASES

Let's sort the customer database, CUST. If dBASE III PLUS is not active, you should start the program, set the default drive, and set the editing menus on. In order to sort a database file you must have an active database file:

1. Position the selection bar on Set Up.
2. Position the selection bar on Database file and press <Return>.
3. Position the selection bar on B: (or A: for hard disk) and press <Return>.
4. Position the selection bar on CUST.DBF and press <Return>.
5. At the prompt, Is the file indexed?, press <Return>.

INTERACTIVE MODE (from the dot prompt):

Type: .USE CUST <Return>

6. Position the selection bar on Organize.
7. Position the selection bar on Sort and press <Return>.
8. Position the selection bar on the field name CUST and press <Return>.
9. Press <Right Arrow> or <Left Arrow> to leave the Fields menu.
10. Position the selection bar on B: (or A: for hard disk) and press <Return>.

11. At the prompt, Enter the name of the file:, type CUSTSORT.
Notice the command you have now built:

SORT ON CUST TO B:(or A:)CUSTSORT

INTERACTIVE MODE (from the dot prompt):

Type: .SORT ON CUST TO CUSTSORT <Return>

12. Now repeat these steps for the database file, LIMO.DBF, and sort it on the CUST field to a file named, LIMOSORT.

DO: SELECT THE TARGET DATABASE WORK AREA

Since CUSTSORT is the file to be updated (the target database), make it active in the currently selected work area:

Press <Esc> to leave Assist because you will not be able to use Assist to perform the following commands:

Type: .SELECT 1 <Return>

Type: .USE CUSTSORT <Return>

Now return to the Assist program:

Type: .ASSIST <Return>

DO: PREPARE A REPORT BEFORE UPDATING

Before proceeding, prepare a report of the data before it is updated by creating a report, CUSTBILL, that prints the customer ID, name, rental amount, paid amount, and due amount:

1. Position the selection bar on Create.
2. Position the selection bar on Report and press <Return>.
3. Position the selection bar on B: and press <Return>.
4. At the prompt, Enter the name of the file:, type CUSTBILL.

INTERACTIVE MODE (from the dot prompt):

Type: .CREATE REPORT CUSTBILL <Return>

5. Create the report specifications. This report should print the CUST, TRIM(FIRST)+ " "+TRIM(LAST), RENTAL, PAID, and DUE. Total all amount fields.
6. Position the selection bar on Retrieve.
7. Position the selection bar on Report and press <Return>.
8. Position the selection bar on B: and press <Return>.
9. Position the selection bar on CUSTBILL.FRM and press <Return>.
10. Position the selection bar on Execute the command and press <Return>.
11. At the prompt, Direct the output to the printer? Y/N, type Y.

INTERACTIVE MODE (from the dot prompt):

Type: .REPORT FORM CUSTBILL TO PRINT <Return>

DO: SELECT THE SOURCE DATABASE WORK AREA

Make the source database, LIMOSORT, active in an unselected work area and then set the active work area to CUSTSORT.

Press <Esc> to leave Assist because you will not be able to use Assist to perform the following commands:

Type: .SELECT 2 <Return>

Type: .USE LIMOSORT <Return>

Type: .SELECT 1 <Return>

DO: UPDATE THE TARGET DATABASE FROM THE SOURCE DATABASE

The following command will perform the actual updating by updating the target database, CUSTSORT, from the source database, LIMOSORT, and by replacing the RENTAL field in the CUSTSORT database with the amount of that field plus the amount of the RENTAL field in the LIMOSORT database. In addition, the DUE field in the CUSTSORT database will be replaced with the new RENTAL amount minus the PAID amount:

Type: .UPDATE ON CUST FROM LIMOSORT REPLACE RENTAL WITH
RENTAL+LIMOSORT->RENTAL,DUE WITH RENTAL-PAID <Return>

(This command is wider than the screen width. Just keep typing and let the command wrap-around as the dBASE III PLUS command line can be 254 characters.)

What happens is this. In the first part of the UPDATE command, dBASE III PLUS reads the customer ID from the field CUST in the first record in the file CUSTSORT. Then it scans LIMOSORT sequentially to find a record with a matching customer ID. If it finds a match, it adds RENTAL in LIMOSORT to RENTAL in CUSTSORT.

If dBASE III PLUS finds no match, it reads the next record in CUSTSORT and scans LIMOSORT for a matching customer ID. The process continues in this manner until the records in CUSTSORT have all been updated.

You've seen the next part of the UPDATE command, REPLACE DUE WITH RENTAL-PAID, previously. It replaces the contents of the DUE field with the new difference between what the customer has been billed and what the customer has paid.

DO: CLEAR THE WORK AREA AND DELETE TEMPORARY FILES

Now that the updating is complete, we need to clean up by removing temporary sorted files. What needs to be done is to remove the original customer database, CUST, from the disk and replace it with the new updated customer database, CUSTSORT, and to remove LIMOSORT from the disk, also.

This can be done with the following commands:

Type: .CLOSE DATABASES <Return>

Type: .ERASE CUST.DBF <Return>

Type: .RENAME CUSTSORT.DBF TO CUST.DBF <Return>

Type: .ERASE LIMOSORT.DBF <Return>

LIMOSORT is going to be deleted, and an active or open database file cannot be deleted. CLOSE DATABASES will close or inactivate all databases in all work areas. ERASE CUST.DBF removes the original CUST file from the disk. RENAME CUSTSORT.DBF TO CUST.DBF renames the new, updated customer database, CUSTSORT, to the original customer database file name, CUST. (Notice that CUST.DBF was deleted before the RENAME in order that the original file name could be used again.)

Note that now, CUST, is in order by customer ID because you produced CUSTSORT by sorting the CUST file at the start of the updating procedure. Finally, ERASE LIMOSORT.DBF removes the sorted automobile file from the disk.

Now return to the Assist program:

Type: .ASSIST <Return>

DO: PRINT A REPORT OF THE UPDATED DATABASE FILE

Now, use the report form file, CUSTBILL, to get a new report of the customer database and compare it with the original report to be certain that the database was updated properly:

1. Position the selection bar on Set Up.
2. Position the selection bar on Database file and press <Return>.
3. Position the selection bar on B: (or A: for hard disk) and press <Return>.
4. Position the selection bar on CUST.DBF and press <Return>.
5. At the prompt, Is the file indexed?, press <Return>.
6. Position the selection bar on Retrieve.
7. Position the selection bar on Report and press <Return>.
8. Position the selection bar on B: (or A: for hard disk) and press <Return>.
9. Position the selection bar on CUSTBILL.FRM and press <Return>.
10. Position the selection bar on Execute the command and press <Return>.
11. At the prompt, Direct the output to the printer? [Y/N], type Y.

INTERACTIVE MODE (from the dot prompt):

Type: .USE CUST <Return>

Type: .REPORT FORM CUSTBILL TO PRINT <Return>

D0: DBASE III PLUS POWER-OFF PROCEDURE

Position the selection bar to **Set Up**, position the selection bar on **Quit dBASE III PLUS**, and then press **<Return>**.

INTERACTIVE MODE (from the dot prompt):

Type: .QUIT <Return>

EXERCISES

1. PREDICTING COLLEGE PERFORMANCE

You have been hired by the Registrar of Sooner University to perform a computer analysis of student data. The student data is contained in two files. You will prepare reports from data in the two student files. The Registrar wants to use the reports to predict student college performance from high school grades and SAT scores.

The data needed to create the reports is in two student files, HIGH and COLL.

Sample data for the HIGH file:

NAME	SOC.SEC.NO.	GPA	SAT	HIGH SCHOOL
ANDERSON, JOHN	123-43-9999	3.21	751	BENTON
ANDERSON, MARYANN	429-82-3198	2.09	459	JOHNSON
BENSON, BONNY	321-87-2974	2.50	500	JOHNSON
CRENSHAW, RENNY	475-61-4872	3.67	789	BENTON
CROKER, JOHN	721-08-8274	3.19	639	SMITH
DAWSON, PETER	274-99-0372	3.03	648	BENTON
DAYTON, ANGELA	582-03-5921	3.78	723	CLARKE
ELLIS, ANITA	714-12-3344	3.45	672	FRANKLIN
FLUTIE, DAN	497-38-3329	4.00	800	FRANKLIN
GRANT, SALLY	429-78-5719	2.96	635	BENTON
HARRIS, DANIEL	920-57-7449	3.22	624	BENTON
HENDERSON, LYNN	345-91-9234	3.72	759	FRANKLIN
JACKSON, MYRA	932-09-9383	3.29	599	BENTON
JACKSON, RAYMOND	450-83-7234	3.97	619	JOHNSON
JOHNSON, DOROTHY	587-98-9276	3.54	678	JOHNSON
JOHNSON, SAMUEL	498-28-3837	3.95	789	FRANKLIN
KANE, BETTY	211-92-3398	3.67	699	SMITH
LONGFELLOW, HENRY	132-32-2487	3.59	721	JOHNSON
MANLOW, BERTHA	782-28-7486	3.54	688	SMITH
NEWMAN, CANDY	989-92-8273	3.11	640	BENTON
OLSON, SWEDE	820-29-8376	2.67	511	JOHNSON
PETERSON, SYLVIA	487-73-4719	2.79	601	SMITH
PETERSON, THOMAS	220-38-5847	2.99	597	CLARKE
RANSOM, HENRY	731-19-9384	3.39	679	CLARKE
RATH, DORA	459-83-7678	3.99	798	FRANKLIN
SAMUELS, DEBBIE	573-48-5857	3.87	688	BENTON
THOMAS, KATHY	411-37-8732	3.07	628	BENTON
VICTOR, VICTORIA	323-37-4756	3.46	674	BENTON
WARNER, CLIFF	411-55-3887	3.72	755	FRANKLIN
ZUSE, CONRAD	411-32-8476	4.00	800	FRANKLIN

Sample data for the COLL file:

NAME	SOC.SEC.NO.	TOTAL HOURS	GRADE POINTS	CUM. GPA	MAJOR
ANDERSON, JOHN	123-43-9999	120	429	3.58	SCIENCE
ANDERSON, MARYANN	429-82-3198	32	58	1.83	FORESTRY
BENSON, BONNY	321-87-2974	67	159	2.38	ENGINEERING
CRENSHAW, RENNY	475-61-4872	85	315	3.71	FORESTRY
CROKER, JOHN	721-08-8274	54	175	3.25	EDUCATION
DAWSON, PETER	274-99-0372	45	144	3.21	EDUCATION
DAYTON, ANGELA	582-03-5921	102	354	3.48	SCIENCE
ELLIS, ANITA	714-12-3344	95	318	3.35	ENGINEERING
FLUTIE, DAN	497-38-3329	90	360	4.00	EDUCATION
GRANT, SALLY	429-78-5719	33	96	2.91	EDUCATION
HARRIS, DANIEL	920-57-7449	72	217	3.02	FORESTRY
HENDERSON, LYNN	345-91-9234	86	325	3.79	SCIENCE
JACKSON, MYRA	932-09-9383	28	75	2.68	AGRICULTURE
JACKSON, RAYMOND	450-83-7234	65	253	3.90	SCIENCE
JOHNSON, DOROTHY	587-98-9276	110	364	3.31	EDUCATION
JOHNSON, SAMUEL	498-28-3837	87	330	3.80	SCIENCE
KANE, BETTY	211-92-3398	58	191	3.31	FORESTRY
LONGFELLOW, HENRY	132-32-2487	94	210	2.24	AGRICULTURE
MANLOW, BERTHA	782-28-7486	86	325	3.79	EDUCATION
NEWMAN, CANDY	989-92-8273	75	225	3.00	AGRICULTURE
OLSON, SWEDE	820-29-8376	46	97	2.12	SCIENCE
PETERSON, SYLVIA	487-73-4719	38	84	2.22	ENGINEERING
PETERSON, THOMAS	220-38-5847	36	87	2.44	AGRICULTURE
RANSOM, HENRY	731-19-9384	42	131	3.13	EDUCATION
RATH, DORA	459-83-7678	56	186	3.33	SCIENCE
SAMUELS, DEBBIE	573-48-5857	72	238	3.31	AGRICULTURE
THOMAS, KATHY	411-37-8732	66	188	2.85	SCIENCE
VICTOR, VICTORIA	323-37-4756	21	68	3.27	SCIENCE
WARNER, CLIFF	411-55-3887	98	370	3.78	EDUCATION
ZUSE, CONRAD	411-32-8476	55	220	4.00	ENGINEERING

The Grade Point Average (GPA) is calculated by dividing the total points accumulated by the total hours taken.

(1) CREATE AND UPDATE THE DATABASES

Create the databases, HIGH and COLL, and add the high school and college data to the databases. Create the HIGH database first and append the data to the file. Then create the COLL database using the same field names for the student name and social security number as in the HIGH database. To get this field information in the COLL database file: .USE COLL then .APPEND FROM HIGH.

After all data has been added to the databases, obtain a print of the database structures by displaying the structure information to the printer (.DISPLAY STRUCTURE TO PRINT).

(2) STUDENT LIST WITH HIGH SCHOOL AND COLLEGE GPA's

Prepare a report with columns for student name, high school GPA, SAT score, and college GPA. The report should be in alphabetical order by student name using the report generation features of dBASE III PLUS. The report should print the required information in a neat and readable form with appropriate column and page headings. Set relations between files to social security number, not student name.

Then print the report in parts which list students by college GPA ranges. The ranges are: 3.50 and above (students with A average), 3.00 to 3.49 (students with B average), 2.50 to 2.99 (students with C average), 2.00 to 2.49 (students with D average), and students below 2.0 (students failing).

(3) STUDENT LIST BY COLLEGE MAJOR

Prepare a report ranking students by SAT score from highest score to lowest score. The report should include the student name, high school name, SAT score, and college major. The report should print the required information in a neat and readable form with appropriate column and page headings. Set relations between files to social security number, not student name.

(4) PREDICTING COLLEGE PERFORMANCE RANKED BY HIGH SCHOOL GPA

Prepare a report analyzing the difference between high school GPA and college GPA. The report should include the student name, high school GPA, college GPA, and the difference between high school GPA and college GPA (high school GPA minus college GPA). The report should print the required information in a neat and readable form with appropriate column and page headings. Set relations between files to social security number, not student name.

The report should group students by high school and should rank them from highest high school GPA to lowest high school GPA.

2. CALCULATING AND UPDATING GRADE POINT AVERAGES

At the end of each semester the semester hours and semester grade points for each student are entered into a database file named SEM. The semester grade point average is calculated to determine who is on the Dean's Honor Roll and then these semester grade points and hours are updated to the COLL database file and the cumulative grade point average in that file is re-calculated.

Sample data for the SEM file:

<u>SOC.SEC.NO.</u>	<u>SEM. HOURS</u>	<u>SEM. POINTS</u>	<u>SEM. GPA</u>
123-43-9999	15	45	
321-87-2974	12	30	
475-61-4872	14	35	
721-08-8274	16	34	
274-99-0372	12	34	
582-03-5921	12	38	
714-12-3344	13	34	
497-38-3329	10	35	
920-57-7449	18	50	
345-91-9234	15	52	
932-09-9383	15	45	
450-83-7234	16	32	
587-98-9276	14	52	
498-28-3837	18	45	
211-92-3398	14	28	
132-32-2487	12	36	
782-28-7486	10	27	
989-92-8273	12	38	
487-73-4719	16	40	
731-19-9384	16	32	
459-83-7678	14	40	
573-48-5857	13	44	
323-37-4756	13	39	
411-55-3887	11	33	
411-32-8476	15	60	

(1) CREATE AND UPDATE THE DATABASES

Create the database, SEM, and add the semester grades data to the database.

After all data has been added to the database, obtain a print of the database structures by displaying the structure information to the printer (.DISPLAY STRUCTURE TO PRINT).

(2) UPDATE SEMESTER GRADE INFORMATION AND PRINT HONOR ROLL

Next, calculate the semester GPA for all the students by dividing the semester hours into the semester points and replacing the semester GPA with the quotient. Prepare a Dean's Honor Roll which includes all students with a semester GPA of 3.25 or greater. This report should include the student name, social security number, major, and semester GPA.

(3) UPDATE SEMESTER GRADES TO COLL DATABASE AND PRINT REPORT

Finally, update the semester grade information to the COLL database file by adding semester hours to total hours accumulated, adding semester grade points to total grade points accumulated, and then re-calculating the cumulative GPA. Prepare a report of the updated COLL database file for all students in alphabetical order which includes the student name, social security number, total hours, grade points, and GPA.

LESSON 11

INPUTTING AND OUTPUTTING TO OTHER PROGRAMS

OBJECTIVES:

- * Outputting to other programs using the COPY TO command
- * ASCII text file formats (SDF and DELIMITED)
- * Transferring data to a text file for mailmerging
- * Creating a word processing master document for mailmerging
- * Inputting from other programs using the APPEND FROM command
- * Transferring data from a spreadsheet program to a database
- * Importing from PFS files (optional)
- * Exporting to PFS files (optional)

REQUIRED MATERIALS:

- * Paper and pen or pencil
- * dBASE III PLUS System Diskettes 1 and 2
- * WordStar with MailMerge or other word processing program
- * Lotus 1-2-3 electronic spreadsheet program
- * PFS:FILE file management program (optional)
- * A practice diskette
- * IBM PC/XT, PS/II, or compatible with 2 disk drives or 1 disk drive and hard disk, and printer

READ: OUTPUTTING TO OTHER PROGRAMS

Data may be transferred from dBASE III PLUS database files to other programs such as PFS files and various mail merge programs through the DELIMITED and SDF (System Data Format) options of the COPY command.

To output dBASE III PLUS database files in a form that can be used by other programs, you must be familiar with the form required by the other program. Also, the other program must be able to accept data in one of the two forms that dBASE III PLUS can produce: DELIMITED or SDF (System Data Format). Most programs have some means for using data files that are in this ASCII text file format.

READ: THE COPY COMMAND AND SYSTEM DATA FORMAT

COPY duplicates all or part of the active database file to a designated target file. All fields will be copied to the target file unless otherwise specified by the FIELDS option. The SDF and DELIMITED options output to a text file with a standard data format and the extension will be .txt. The scope of the COPY can be qualified by using the FOR option before the SDF or DELIMITED option.

The following example copies the named fields, Last, Address, City, State, and Zip from the active database, Refnames, to the target data file, Mail:

```
.USE Refnames
.COPY FIELDS L name,Addr,City,St,Zip TO Mail SDF
      10 Records Copied
```

An example of the data file created by this COPY command with the SDF option would look something like this:

Neuhoff	8653 Rita Drive	Bloomington	IN47401
Lakeland	4902 Bluffside Road	Muskeogen	MI49450
Butler	45 Macedonia Road	Raleigh	NC2760
Roarke	87899 Gallatin SW	Roanoke	VA24018
Randolph	894 Grigsby Road	Knoxville	TN37922
Greystroke	8897 Plaines Lane	Cheyenne	WY82005
Lock	5788 Certified Way	Carson City	NV89701
Campbell	2716 Etoile Way	LaFayette	KY42254
Lyman	78 Rye Street	Topeka	KS66699

READ: MAIL MERGE PROGRAMS AND DELIMITED FILES

Mail merge programs are popularly used with word processing programs to use data from a data text file and merge all or part of that data with information in a form letter. Generally, mail merge programs use files in which the records are delimited (separated) by a carriage return and a line feed, and the fields are delimited by commas. Some fields may be further delimited by double quotation marks in addition to the commas.

To create this type of file from a dBASE III PLUS database file, the COPY command with the DELIMITED option is used.

```
.USE Refnames
.COPY TO Mail DELIMITED WITH "
      10 Records Copied
```

An example of the mailing list data file created by the COPY command and delimited with double quotation marks would look something like this:

```
"Neuhoff","Luann","8653 Rita Drive","Bloomington","IN","47401"  
"Lakeland","Larry","4902 Bluffside Road","Muskegon","MI","49450"  
"Butler","Sarah","45 Macedonia Road","Raleigh","NC","27606"  
"Roarke","John","87899 Gallatin SW","Roanoke","VA","24018"  
"Randolph","Charles","894 Grigsby Road","Knoxville","TN","37922"  
"Greystroke","Arthur","8897 Plains Lane","Cheyenne","WY","82005"  
"Lock","Joseph","5788 Certified Way","Carson City","NV","89701"  
"Campbell","Chester","2716 Etoile Way","Lafayette","KY","42254"  
"Lyman","William","78 Rye Street","Topeka","KS","66699"
```

READ: TRANSFERRING TO A DATA FILE FOR MAILMERGING

You wish to prepare statements for all your customers. In order to do this you must transfer the data from the CUST database file to a data file that can be used with a merge program and merge this file with a word processing master document.

Let's accomplish this by transferring the CUST database file to a data text file.

DO: SELECT A DATABASE FILE

If dBASE III PLUS is not active, you should start the program, set the default drive, and set the editing menus on. In order to transfer the data in CUST to a data file, it must be the active database:

1. Position the selection bar on **Set Up**.
2. Position the selection bar on **Database file** and press **<Return>**.
3. Position the selection bar on **B:** (or **A:** for hard disk) and press **<Return>**.
4. Position the selection bar on **CUST.DBF** and press **<Return>**.
5. At the prompt, **Is the file indexed?**, press **<Return>**.

INTERACTIVE MODE (from the dot prompt):

Type: **.USE CUST <Return>**

DO: OUTPUT TO A DELIMITED DATA FILE FORMAT

Now, output to a delimited data file format. To make this file more general purpose and easy to use with most other programs, we are going to delimit it with quotation marks. Also, we will copy all fields to the data file:

Press <Esc> to leave Assist because you will not be able to use Assist to perform the COPY TO command from the Assistant:

Type: .COPY TO CUST DELIMITED WITH " <Return>

You now have the customer data stored in data text file format delimited with quotation marks in the file, CUST.TXT.

DO: DBASE III PLUS POWER-OFF PROCEDURE

You are going to quit dBASE III PLUS and start your word processing program with a mail merge function. Since you are already at the dot prompt, just type the following command to quit dBASE III PLUS:

Type: .QUIT <Return>

DO: CREATE A MASTER DOCUMENT

At this point you should start your word processing program. Be sure your word processing program has a merge function with which you are familiar and that the merge function uses data from standard ASCII text files.

WORDSTAR EXAMPLE:

The following example will use the WordStar word processing program to create a master document to use with MailMerge.

Type the following document into a WordStar file named, STATEMNT.MST using the appropriate merge commands as shown:

.op
.df cust.txt
.rv custid,last,first,address,city,state,zip,rental,paid,due
.sv date, April 25, 1987

AL'S LIMOUSINE SERVICE

S T A T E M E N T

Customer No.: &custid&

&last&, &first&

&address&

&city&, &state& &zip&

For Rentals due as of &date&:

Amount Billed to Date: \$&billed&

Amount Paid to Date: \$&paid&

Current Amount Due: \$&due&

.pa

OTHER WORD PROCESSING PROGRAM

Your instructor may request you to use another word processing program other than WordStar. At this point you should load the requested word processing program and type the master document shown above using the file name, STATEMNT.MST and using appropriate merge commands in place of the WordStar dot commands and &variables.

DO: MERGE THE DATA FILE WITH THE MASTER DOCUMENT

Now use the merge function of your word processing program to merge the data in the data file with the information in the form letter to produce statements for each customer.

WORDSTAR/MAILMERGE EXAMPLE:

The following steps explain how to use the MailMerge function of WordStar to merge the master document with the customer data file (you must have MailMerge on your WordStar disk):

At the WordStar Opening Menu:

Type: M

You see the prompt:

Name of file to MailMerge?

Type: STATEMENT.MST

Ready the printer and press <Esc>.

You should have a statement for each customer printed out. After several of the statements have been printed you can stop the print:

Type: P

You see the prompt:

"Y" to abandon print, "N" to resume, ^U to hold

Type: Y <Return>

Press X to exit WordStar and start dBASE III PLUS, set the default drive, and set the editing menus on.

OTHER WORD PROCESSING MERGE PROGRAM

Your instructor may request you to use another word processing program other than WordStar. At this point you should load the requested word processing program and perform the appropriate merge commands. Be certain that your word processing program accepts a merge file in delimited format.

Exit your word processing/merge program and start dBASE III PLUS, set the default drive, and set the editing menus on.

DO: EXPORTING TO PFS FILES (OPTIONAL)

The Export function converts dBASE III PLUS database files to files you can use with PFS, also converting the dBASE format file to the PFS screen form. The EXPORT command is available from the Assistant menu while the COPY TO command is not.

Create a database file named **MAIL** and data entry screen named **MAILFORM** for the following database structure:

<u>Field Name</u>	<u>Type</u>	<u>Width</u>	<u>Decimals</u>
L_NAME	C	10	
F_NAME	C	10	
ADDRESS	C	20	
CITY	C	20	
STATE	C	2	
ZIP_CODE	N	5	0

Now append the following data to the database using the data entry screen you have created:

Neuhoff, Luann, 8653 Rita Drive, Bloomington, IN, 47401
Lakeland, Lionel, 4902 Bluffside Road, Muskegon, MI, 49450
Butler, Sarah, 45 Macedonia Road, Raleigh, NC, 27604
Roarke, John, 87899 Gallatin SW, Roanoke, VA, 24018
Randolph, Charles, 894 Grigsby Road, Knoxville, TN, 37922
Greystroke, Arthur, 8897 Plaines Lane, Cheyenne, WY, 82005
Lock, Joseph, 5788 Certified Way, Carson City, NV, 89701
Campbell, Chester, 2716 Etoile Way, LaFayette, KY, 42254
Lyman, William, 78 Rye Street, Topeka, KS, 66699

To convert a dBASE III PLUS database and screen files to a PFS file, you use the **Export** option of the **Tools** menu:

1. Position the selection bar on **Set Up**.
2. Position the selection bar on **Database file** and press **<Return>**.
3. Position the selection bar on **B:** (or **A:** for hard disk) and press **<Return>**.
4. Position the selection bar on **MAIL.DBF** and press **<Return>**.
5. At the prompt, **Is the file indexed?**, press **<Return>**.
6. Position the selection bar on **Format** and press **<Return>**.
7. Position the selection bar on **B:** (or **A:** for hard disk) and press **<Return>**.
8. Position the selection bar on **MAILFORM.FMT** and press **<Return>**.
9. Position the selection bar on **Tools**.
10. Position the selection bar on **Export** and press **<Return>**.

11. Position the selection bar on B: (or A: for hard disk) and press <Return>.
12. At the prompt, Enter the name of the file:, type MAIL and press <Return>.

Notice the following command on the command line:

EXPORT TO B:(or A:)MAIL TYPE PFS

The status bar keeps track of the records converted. A PFS file with the name, MAIL, is created.

INTERACTIVE MODE (from the dot prompt):

Type: .EXPORT TO B:(A:) MAIL TYPE PFS <Return>

13. You can now use the converted file, provided you also have a copy of PFS:FILE. If you have the PFS:FILE program, look at the data in the new file to see how it appears.
14. Exit PFS and start dBASE III PLUS, set the default drive, and set the editing menus on.

READ: INPUTTING FROM OTHER PROGRAMS

Inputting data into dBASE III PLUS database files from other programs such as Lotus 1-2-3 is accomplished through the DELIMITED and SDF (System Data Format) options of the APPEND FROM command. To transfer data created from an external program into a dBASE III PLUS database file, the other program must be able to output data in one of the two forms dBASE III PLUS can accept: DELIMITED or SDF.

If a file extension to be APPENDED FROM is not entered as part of the file name and the SDF/DELIMITED option is not specified, dBASE III PLUS will assume a .dbf extension. If a file extension is not entered as part of the file name, and the SDF/DELIMITED option is specified, dBASE III PLUS will assume a .txt extension.

If the SDF option is used, data will be copied character for character beginning from the left. The end of an SDF record is indicated by a carriage return/line feed.

If the DELIMITED option is used, data will be copied on a field by field basis beginning from the left. The end of a field is indicated by a comma. The end of a delimited record is indicated by a carriage return/line feed.

To illustrate the transfer of SDF data into a dBASE III PLUS database file, assume that a program has created the following SDF file:

Neuhoff	Luann	8653 Rita Drive	Bloomington	IN47401
Lakeland	Lionel	4902 Bluffside Road	Muskegen	MI49450
Butler	Sarah	45 Macedonia Road	Raleigh	NC27604
Roarke	John	87899 Gallatin SW	Roanoke	VA24018
Randolph	Charles	894 Grigsby Road	Knoxville	TN37922
Greystroke	Arthur	8897 Plaines Lane	Cheyenne	WY82005
Lock	Joseph	5788 Certified Way	Carson City	NV89701
Campbell	Chester	2716 Etoile Way	LaFayette	KY42254
Lyman	William	78 Rye Street	Topeka	KS66699

To transfer this file into dBASE III PLUS data base, a structure must be created where the field widths match (in order) the number of characters in each of the columns in the above SDF file.

A database structure that matches this file would look like this:

Field Name	Type	Width	Decimals
L_NAME	C	10	
F_NAME	C	10	
ADDRESS	C	20	
CITY	C	20	
STATE	C	2	
ZIP_CODE	N	5	0

SDF files such as the one above are often referred to as flat files. The data is transferred into the dBASE III PLUS database file by use of the APPEND FROM command:

.APPEND FROM filename SDF

READ: INPUTTING A FILE FROM A SPREADSHEET

You keep a check register for Al's Limousine Service using the electronic spreadsheet program, Lotus 1-2-3. For each check that is written, the check number, description, check amount, expense category, and balance forward is recorded. You want to combine all the checks by expense category, subtotal the expense category, and print a report.

The spreadsheet program is not effective at performing the subtotalling, so you decide to transfer the data to a dBASE III PLUS database file and use the report generator.

DO: ENTER THE SPREADSHEET DATA

Let's enter the spreadsheet data and transfer the worksheet to a database file. If Lotus 1-2-3 is not active, you should start the program. Enter the following spreadsheet data using the Worksheet, Column-Width (/WCS) command to set the following column widths: A=5, B=25, C=12, D=9. Set two decimal places in the last column with the Range Format (/RFF2 D7.D30) command. Enter a formula to total the check amounts in cell D30 as @SUM(D7.D28):

	A	B	C	D	E
1	Al's Limousine Service				
2	Checks for August, 1985				
3					
4	Check				
5	No.	Description	Category	Amount	
6	-----				
7	102	Oscar's Office Supplies	Supplies	58.60	
8	103	Raymond Gunn Security	Security	1100.00	
9	104	S.H. Eister, Atty.	Legal	600.00	
10	105	Eternal Gas and Elec.	Utility	185.60	
11	106	Al Fixit	Payroll	650.00	
12	107	Lynn Rental	Payroll	575.00	
13	108	Sue Secretary	Payroll	435.00	
14	109	Ding Dong Bell	Telephone	385.45	
15	110	Auto World Weekly	Advertising	650.00	
16	111	Auto Mechanics, Inc.	Maintenance	837.50	
17	112	Oscar's Office Supplies	Supplies	120.56	
18	113	Sue Themfirst, Atty.	Legal	275.00	
19	114	Rent-All Property Mngmt.	Rent	1500.00	
20	115	Car-Search Monthly	Advertising	450.00	
21	116	Clean-Up Office Serv.	Maintenance	300.00	
22	117	Fill-Er-Up Stations	Fuel and Oil	175.00	
23	118	Joe Programmer	Consulting	300.00	
24	119	Fill-Er-Up Stations	Fuel and Oil	195.00	
25	120	Al Fixit	Payroll	650.00	
26	121	Lynn Rental	Payroll	575.00	
27	122	Sue Secretary	Payroll	435.00	
28	123	Joe Programmer	Consulting	300.00	
29	-----				
30	TOTAL FOR MONTH				
31					=====

DO: PRINT THE SPREADSHEET

After you have completed the worksheet, save it in a file named CHECKS:

Type: /FS CHECKS

Now print the worksheet to the printer:

Type: /PP

Set the print range to include the complete worksheet and then quit the print to printer function:

Type: R A1.D31 <Return>

Type: G

Type: Q

DO: CREATE A TEXT FILE

Now print the worksheet to a text file named CHECKS (Lotus 1-2-3 will automatically add the extension, .PRN):

Type: /PF CHECKS

Set the print range to include only the check data (no titles, headings, or totals):

Type: R A7.D28 <Return>

Reset the left and top margin options to zero:

Type: OML 0 <Return>

Type: MT 0 <Return>

Reset header and footer options by removing all reserved header and footer space and then quit the options:

Type: OU

Type: Q

Now, print the file to a text file on the disk and then quit the print to a file function:

Type: G

Type: Q

You now have a SDF text file stored on the disk which can be accessed by dBASE III PLUS. Exit Lotus 1-2-3:

Type: /OY

Type: EY

DO: CREATE A DATABASE FILE

Start the dBASE III PLUS program, set the default drive, and set the editing menus on. Now create a database to which the text file, CHECKS.PRN can be input:

1. Position the selection bar on Create.
2. Position the selection bar on Database file and press <Return>.
3. Position the selection bar on B: (or A: for hard disk) and press <Return>.
4. Enter the name of the file as CHECKS.

INTERACTIVE MODE (from the dot prompt):

Type: .CREATE CHECKS <Return>

Use the following information for the database structure:

<u>Field Name</u>	<u>Type</u>	<u>Width</u>	<u>Decimals</u>
CHECK_NO	N	5	
DESC	C	25	
CATEGORY	C	12	
AMOUNT	N	9	2

Notice that the database widths match the widths of the columns in the worksheet.

DO: SELECT A DATABASE FILE

1. Position the selection bar on Set Up.
2. Position the selection bar on Database file and press <Return>.
3. Position the selection bar on B: (or A: for hard disk) and press <Return>.

INPUTTING AND OUTPUTTING TO OTHER PROGRAMS 11-12

4. Position the selection bar on **CHECKS.DBF** and press **<Return>**.
5. At the prompt, **Is the file indexed?**, press **<Return>**.

INTERACTIVE MODE (from the dot prompt):

Type: **.USE CHECKS <Return>**

DO: INPUTTING TO A DATABASE

To input the text file into the database use the **APPEND FROM** command qualified with the **SDF** option:

Press **<Esc>** to leave Assist because you will not be able to use Assist to perform the **APPEND FROM** command:

Type: **.APPEND FROM CHECKS.PRN SDF <Return>**

Return to the Assist program to complete this exercise:

Type: **.ASSIST <Return>**

DO: INDEX THE DATABASE

Now index the data by **CATEGORY** and **USE** the **CHECKS** database file with the new index:

1. Position the selection bar on **Organize**.
2. Position the selection bar on **Index** and press **<Return>**.
3. Press **<F10>** to display the Fields menu on the screen.
4. Position the selection bar on the field name **CATEGORY** and press **<Return>** and then press **<Return>** again in response to the completed prompt.
5. Position the selection bar on **B:** (or **A:** for hard disk) and press **<Return>**.
6. At the prompt, **Enter the name of the file:**, type **CHECKCAT**.

Notice the command you have now built:

INDEX ON CATEGORY TO B:(or A:)CHECKCAT

7. Now position the selection bar on Set Up.
8. Position the selection bar on Database file and press <Return>.
9. Position the selection bar on B: (or A: for hard disk) and press <Return>.
10. Position the selection bar on CHECKS.DBF and press <Return>.
11. At the prompt, Is the file indexed?, type Y.
12. Position the selection bar on CHECKCAT.NDX and press <Return>. Notice the command you have now built:

USE B:(or A:)CHECKS INDEX CHECKCAT

13. Press <Right Arrow> or <Left Arrow> to leave the Index menu and USE the database as an indexed database.

INTERACTIVE MODE (from the dot prompt):

Type: .INDEX ON CATEGORY TO CHECKCAT <Return>

Type: .USE CHECKS INDEX CHECKCAT <Return>

DO: CREATE AND PRINT A REPORT

Create a report, CHECKRPT1, that includes all the fields in the database and subtotals the check database file by category. Total the check amount field:

1. Position the selection bar on Create.
2. Position the selection bar on Report and press <Return>.
3. Position the selection bar on B: (or A: for hard disk) and press <Return>.
4. At the prompt, Enter the name of the file:, type CUSTRPT1.

INTERACTIVE MODE (from the dot prompt):

Type: .CREATE REPORT CHECKRPT1 <Return>

5. Now enter the report headings and specifications. Don't forget to specify the report program to subtotal on the CATEGORY field.
6. Now you are ready to print the report. Position the selection bar on Retrieve.
7. Position the selection bar on Report and press <Return>.
8. Position the selection bar on B: (or A: for hard disk) and press <Return>.
9. Position the selection bar on CHECKRPT1.FRM and press <Return>.
10. Position the selection bar on Execute the command and press <Return>.
11. At the prompt, Direct the output to the printer? Y/N, type Y.

INTERACTIVE MODE (from the dot prompt):

Type: .REPORT FORM CHECKRPT1 TO PRINT <Return>

DO: IMPORTING FROM PFS FILES (OPTIONAL)

The Import function converts PFS files to dBASE III PLUS database files and converts the PFS screen form to a dBASE III PLUS format file. The IMPORT command is available from the Assistant menu while the APPEND FROM command is not.

Design a PFS screen and file named MAILLIST which stores the following field information (allow at least the minimum width shown for each field):

<u>Field Name</u>	<u>Width</u>
LAST	10
F NAME	10
ADDRESS	20
CITY	20
STATE	2
ZIP_CODE	5

Now add the following data to the file:

Neuhoff, Luann, 8653 Rita Drive, Bloomington, IN, 47401
Lakeland, Lionel, 4902 Bluffside Road, Muskegon, MI, 49450
Butler, Sarah, 45 Macedonia Road, Raleigh, NC, 27604
Roarke, John, 87899 Gallatin SW, Roanoke, VA, 24018
Randolph, Charles, 894 Grigsby Road, Knoxville, TN, 37922
Greystroke, Arthur, 8897 Plaines Lane, Cheyenne, WY, 82005
Lock, Joseph, 5788 Certified Way, Carson City, NV, 89701
Campbell, Chester, 2716 Etoile Way, LaFayette, KY, 42254
Lyman, William, 78 Rye Street, Topeka, KS, 66699

To convert a PFS file to a dBASE III PLUS database and screen files, you use the **Import** option of the **Tools** menu:

1. Position the selection bar on **Tools**.
2. Position the selection bar on **Import** and press **<Return>**.
3. Position the selection bar on **B:** (or **A:** for hard disk) and press **<Return>**.
4. At the prompt, **Enter the name of the file:**, type **MAILLIST** and press **<Return>**. Notice the following command on the command line:

IMPORT FROM B:(or A:)MAILLIST

The status bar keeps track of the records converted. A dBASE III PLUS database file, **MAILLIST.DBF**, and a screen file, **MAILLIST.FRM**, are created.

INTERACTIVE MODE (from the dot prompt):

Type: **.IMPORT FROM B:(A:)MAILLIST <Return>**

5. Now look at the data in your new database file using the data entry form to see how it appears.

DO: DBASE III PLUS POWER-OFF PROCEDURE

Position the selection bar to **Set Up**, position the selection bar on **Quit dBASE III PLUS**, and then press **<Return>**.

INTERACTIVE MODE (from the dot prompt):

Type: **.QUIT <Return>**

EXERCISES

1. THE TELETHON

The local public television station has recently completed a weekend telethon to raise money for operations for the next fiscal year. Pledges were received over the telephone and information about each caller, including the total amount of the pledge and the rate at which the pledge will be paid, was recorded on a pledge card.

The pledge card information is to be stored on the computer. A master form letter requesting the pledge to be paid is to be created. Then the pledge data should be merged with the master letter to generate a personalized letter of solicitation for the pledge amount.

The following is the master form letter:

K P U B TELEVISION STATION
242 SATELLITE LANE
LACOSTA, CALIFORNIA 94567
(213) 678-4321

April 25, 1986

[Title and Name]
[Address]
[City, State Zip]

Dear [Title and Last Name]:

Thank you for your pledge of [Pledge Amount] made during our recent telethon. As you probably know, the pledges made by our viewers are the financial basis for operating KPUB.

It is because of your financial support that we are able to keep public television on the air in [City].

To pay your [How Often] pledge of [Periodic Amount], you may use the enclosed envelopes. Thank you again, [Title and Last Name], for your continued support of your public broadcasting station, KPUB.

Sincerely,

Gerald E. Free
Station Manager

All expressions enclosed in brackets mark a place in the letter where corresponding information from your pledge data file should be merged.

INPUTTING AND OUTPUTTING TO OTHER PROGRAMS 11-17

The following are some of the pledge cards:

Ms.	Rosalind	West
TITLE	FIRST NAME	LAST NAME
1235 S. Court St.		
ADDRESS		
Circleville	California	98776
CITY	STATE	ZIP CODE
\$120.00	Monthly	\$10.00
TOTAL PLEDGE AMOUNT	HOW OFTEN?	PERIODIC AMOUNT

Dr.	Norman	Long
TITLE	FIRST NAME	LAST NAME
431 Imperial Court		
ADDRESS		
Frisco Bay	California	98779
CITY	STATE	ZIP CODE
\$1000.00	Annual	\$1000.00
TOTAL PLEDGE AMOUNT	HOW OFTEN?	PERIODIC AMOUNT

Mr.	Hector	Gonzales
TITLE	FIRST NAME	LAST NAME
369 Ludwig Dr.		
ADDRESS		
Bernalillo	California	94321
CITY	STATE	ZIP CODE
\$240.00	Monthly	\$20.00
TOTAL PLEDGE AMOUNT	HOW OFTEN?	PERIODIC AMOUNT

Mr.	Gary	Logan
TITLE	FIRST NAME	LAST NAME
5067 Nelson Dr.		
ADDRESS		
Ridgewood	Nevada	92476
CITY	STATE	ZIP CODE
\$500.00	Semi-annual	\$250.00
TOTAL PLEDGE AMOUNT	HOW OFTEN?	PERIODIC AMOUNT

Ms.	Tracy	Allen
TITLE	FIRST NAME	LAST NAME
364 E. Franklin St.		
ADDRESS		
Mesa Verde	Arizona	89431
CITY	STATE	ZIP CODE
\$400.00	Quarterly	\$100.00
TOTAL PLEDGE AMOUNT	HOW OFTEN?	PERIODIC AMOUNT

Ms.	Penny	Anderson
TITLE	FIRST NAME	LAST NAME
124 Caroline Ct.		
ADDRESS		
Ocean View	California	98994
CITY	STATE	ZIP CODE
\$120.00	Monthly	\$10.00
TOTAL PLEDGE AMOUNT	HOW OFTEN?	PERIODIC AMOUNT

(1) CREATE AND UPDATE THE DATABASE

Create a database file named, PLEDGE, and add the pledge card information for all pledges to the database.

After all pledge information has been added to the database, obtain a print of the database structure by displaying the structure information to the printer.
(.DISPLAY STRUCTURE TO PRINT)

Output the database file to a data file format suitable for merging with a form letter.

(2) CREATE A MASTER DOCUMENT AND MAILMERGE

Create a master document using the form letter shown above.

Merge the data file and the form letter to generate personalized form letters. You should have one letter for each person who made a pledge.

2. THE SHOE STORE

A local retail shoe store uses an electronic spreadsheet program to record the daily sales of men's and women's shoes. The owner of the store is interested in finding which brands, styles, and sizes of shoes are selling the most.

You decide that the indexing and report generator features of dBASE III PLUS would make it easier to analyze the data and so you decide to transfer the daily sales information to a database file to analyze it.

Use the following worksheet:

SAM'S SHOE STORE
Daily Sales for August 10, 1986

SALE NO.	BRAND	STYLE	COLOR	SIZE	M/W	COST
115	JASMIN	FLATS	BROWN	6	W	39.95
116	NATURALIZER	FLATS	BROWN	7	W	34.95
117	CONNIE	PUMPS	BLACK	5	W	36.80
118	FOOTWORKS	ATHLETIC	BLUE	4	W	32.95
119	NIKE	ATHLETIC	WHITE	9	M	29.95
120	BANDALINO	HEELS	BLACK	7	W	42.45
121	JARMAN	DRESS	BROWN	8	M	49.95
122	FLORSHEIM	DRESS	BLACK	9	M	45.80
123	COASTERS	SANDAL	BROWN	5	W	29.95
124	DEXTER	CASUAL	BROWN	10	M	34.95
125	STADIA	ATHLETIC	WHITE	7	W	36.95
126	NATURALIZER	FLATS	BROWN	7	W	34.95
127	BANDALINO	HEELS	BROWN	6	W	44.95
128	FLORSHEIM	DRESS	BROWN	9	M	45.80
129	JASMIN	FLATS	BLUE	7	W	39.95
130	FOOTWORKS	ATHLETIC	WHITE	5	W	32.80
131	JASMIN	FLATS	BLACK	7	W	34.95
132	NATURALIZER	PUMPS	BROWN	6	W	38.60
133	FLORSHEIM	CASUAL	BROWN	9	M	36.40
134	NATURALIZER	FLATS	BROWN	5	W	34.95

TOTAL SALES FOR DAY

=====

(1) CREATE AND UPDATE THE WORKSHEET

Create a worksheet file named, SALES, and enter the worksheet as shown above. Total the sales for the day.

After all worksheet information has been added to the worksheet, obtain a print of the worksheet using the Print command of the spreadsheet program.

Output the worksheet file to a text file format suitable for inputting to a database file.

Create a database file named, SALES, and APPEND the text file to the database file. Obtain a print of the database structure by displaying the structure information to the printer (.DISPLAY STRUCTURE TO PRINT).

(2) GENERATE DATABASE REPORTS

Create a report which shows the most popular brands and styles (index and subtotal by BRAND + STYLE). Include all fields on the report and total the amount field.

Create a report which shows the shoe size most widely worn for men and women (index and subtotal by M/W + SIZE). Include all fields on the report and total the amount field.

Create a report which shows the most expensive shoes (sort on AMOUNT descending and subtotal on AMOUNT). Include all fields on the report and total the amount field.

LESSON 12

WRITING SIMPLE PROGRAMS

OBJECTIVES:

- * Explain dBASE III PLUS command files
- * Using the dBASE Word Processor
- * Developing an update program
- * Developing a summary program
- * Developing a report program

REQUIRED MATERIALS:

- * Paper and pen or pencil
- * dBASE III PLUS System Diskettes 1 and 2
- * A practice diskette
- * IBM PC/XT, PS/II, or compatible with 2 disk drives or 1 disk drive and hard disk, and printer
- * Programming with dBASE III PLUS Reference Manual

READ: COMMAND FILES

Frequently, users repeat a series of commands when a procedure or routine becomes established. The need arises then, to have the capacity to store this procedure--or series of commands--in a file to be executed automatically by dBASE III PLUS. This is what a command file does. A command file is another name for a dBASE III PLUS program.

A dBASE III PLUS program is simply a series of commands typed into a text file, similar to a letter which is written with a word processing program. dBASE III PLUS has a built-in word processor or editor that is accessed with the MODIFY COMMAND command. Since this editor is somewhat limited in size and function, many times programmers use an editor such as WordStar or almost any other word processing program to develop dBASE III PLUS programs.

If no extension is used, dBASE III PLUS automatically adds the extension, .prg, to a command file. Command file names must follow the same file naming conventions used for other types of dBASE III PLUS files.

The dBASE III PLUS command, DO, causes a dBASE III PLUS command file to be executed. The file name included with the DO command must include the drive designator if other than the default drive. A .prg extension is assumed unless otherwise specified.

When the program called by a DO is complete, control will be returned to the calling program, or to the keyboard if the DO was issued from the dot prompt. For example:

A program, Taxes.prg, is located on disk drive B. The default drive is A. To execute the program, Taxes.prg:

```
.DO B:Taxes
```

A program, Areacalc, is located on the default drive. This program calculates the area of a rectangle based on the formula $\text{Area} = \text{Length} * \text{Width}$.

To create this program and then execute it, you would type the following command:

```
.MODIFY COMMAND Areacalc
```

The dBASE III Word Processor screen would display and you would type in the following dBASE III PLUS commands to be stored in a command (program) file:

```
SET TALK OFF
STORE 4 TO Length
? "The Length is ",Length
STORE 6 TO Width
? "The Width is ",Width
? "The Area is ",Length * Width
```

You would press <Ctrl> W to save the program. When the dBASE III PLUS prompt displays, you would type the following to execute the command file:

```
.DO Areacalc
```

The program would execute displaying the following information on the screen:

```
The Length is      4
The Width is       6
The Area is        24
```

READ: THE dBASE WORD PROCESSOR

MODIFY COMMAND invokes the dBASE III PLUS word processor. Its main function is to create and edit command files, but it can also be used on any standard ASCII text file. To create or modify a command file named, Calcprog, you would use the following command:

.MODIFY COMMAND Calcprog

If no drive designator or extension is explicitly specified as part of the file name, the default drive and .prg extension are assumed.

When this command is issued, dBASE III PLUS searches for the file. If it exists, it is called up for editing. Each time a file is edited, the previous version is saved as a backup file with a .bak extension. If the file is not found, a new file is created.

To obtain a print out (hard copy) of a file created by MODIFY COMMAND, issue the command TYPE <file name> TO PRINT at the dot prompt.

The maximum file size that the dBASE word processor can handle is 5,000 characters--approximately 100 to 200 lines.

The following is a list of keys that have special functions in MODIFY COMMAND (word processor) mode:

Arrow keys	-	Move the cursor one position in the indicated direction.
Backspace	-	Deletes the character to the left of the cursor.
^A, <Home>	-	Moves the cursor to the beginning of the current or the previous word.
^C, <PgDn>	-	Scrolls the screen down 18 lines.
^D	-	Equivalent to the right arrow key.
^E	-	Equivalent to the up arrow key.
^F, <End>	-	Moves the cursor to the beginning of the next word.
^G, <Del>	-	Deletes the character at the cursor.
^KR	-	Reads another file into the file being edited at the cursor position.
^KW	-	Writes the entire file to another file.
^M, <Return>	-	Moves the cursor to the beginning of the next line.
^N	-	Inserts a blank line after the cursor.
^Q, <Esc>	-	Exits without saving changes.
^R, <PgUp>	-	Scrolls the screen up 18 lines.
^S	-	Equivalent to the left arrow key.

- ^T** - Deletes all characters from the current cursor position to the beginning of the next word.
- ^V, <Ins>** - Toggles between the insert and overwrite modes.
- ^W, ^<End>** - Exits and saves changes.
- ^X** - Equivalent to the down arrow key.
- ^Y** - Deletes the current line.

READ: AN UPDATE PROGRAM

From time to time Al's Limousine Service customers send in payments. The process to record the customer payment is fairly simple. For example, if customer 2308 sent a payment of \$1200.00 the following commands could be entered assuming CUST is active:

```
.REPLACE PAID WITH PAID + 1200 FOR CUST = 2308
.REPLACE DUE WITH DUE - 1200 FOR CUST = 2308
```

It's a common practice to accumulate payment data for a few days and then enter it all at once. The process could be improved by saving the commands as a program in a command file. This program would ask for the customer ID and amount and would update the corresponding customer records automatically.

A command file is nothing more than a list of commands you could enter directly from the keyboard. If you can figure out how to do it manually, as with the two commands above, you can write a command file to do the work automatically.

DO: DEVELOPING AN UPDATE PROGRAM

If dBASE III PLUS is not active, you should start the program, set the default drive, and set the editing menus on. Since we are not going to be using the Assistant program for this unit, press <Esc> to leave Assist.

To enter a command file named UPDATE in the dBASE word processor:

```
Type: .MODIFY COMMAND UPDATE <Return>
```

When dBASE Word Processor appears on the top line of the screen, begin typing in the program as shown below. The word processor works like any word processing program and has many of the same control key commands as WordStar. Refer to the section, The dBASE III PLUS Word Processor, for a full explanation of control key commands.

As each line is typed, be certain to press <Return> at the end of the line. After typing the program, press, <Ctrl> W to save the command file:

```
*program name: update.prg      author: joe programmer
*this program records and updates customer payments
*files used: cust.dbf index custno.ndx

USE cust INDEX custno

CLEAR
?
?" UPDATE CUSTOMER FILE WITH AMOUNTS RECEIVED AS PAYMENTS"
?
INPUT "Enter Customer ID, or 0 to Quit " TO mnumber
DO WHILE mnumber <> 0
    ?
    SEEK mnumber
    DISPLAY first,last,due
    ?
    INPUT "Enter Amount Paid " TO mamount
    REPLACE paid WITH paid + mamount, due WITH due - mamount
    ?
    DISPLAY first,last,due
    ?
    INPUT "Enter Customer ID, or 0 to Quit " TO mnumber
ENDDO
CLEAR
?
?"                               END OF UPDATE PROGRAM"
CLOSE DATABASES
RETURN
```

To save the program in a command file:

Type: <Ctrl> W

READ: DBASE III PLUS PROGRAMMING CONVENTIONS

A few dBASE III PLUS programming conventions are used that you should notice:

- o All dBASE III PLUS keywords are in uppercase while all other words are in lower case.
- o Indentation and skipping lines are used to set off logic structures and make the program more readable.
- o Comments are used at the beginning of the program to provide information about the program including program name, author, description, files used, and date of last update.
- o Memory variables are usually preceded by the letter, m.

WRITING SIMPLE PROGRAMS 12-5

READ: HOW THE PROGRAM WORKS

This program begins by clearing the screen, displaying a message identifying the purpose of the program, and then asking for the customer ID number and the payment amount. It will then update both fields in the appropriate record in the CUST file. It will also display the name of the customer and the amount owed before the payment and after the payment. It will continue to receive input data until the operator enters a zero for the customer ID prompt. It will then display the message: END OF UPDATE PROGRAM.

With a few exceptions the program consists of commands you have already used. The following is a step-by-step explanation of how the program works, including an explanation of the new dBASE III PLUS keywords:

- A. The program executes sequentially line by line from top to bottom.
- B. The asterisks at the beginning of a line cause dBASE III PLUS to ignore that line so that it is not executed. These are comment lines only for the use of the programmer.
- C. The CLEAR command clears the screen and positions the cursor to the upper left-hand corner.
- D. The INPUT command prompts the user for keyboard entry until a carriage return is entered. The character string inside the quotation marks will be displayed on the screen and then dBASE III PLUS will wait until the operator enters data and presses <Return>. The data entered is stored in the numeric memory variable following the T0 phrase--in this case, mnumber. To input data into a character memory variable, use ACCEPT in place of INPUT.
- E. The DO WHILE command works in conjunction with the ENDDO command to create a loop (a sequence of statements executed repetitively). These commands define the starting point and ending point of the loop. The loop continues to execute (repeat) as long as the logical condition stated on the DO WHILE command remains true. When the condition becomes false by entering a zero for the customer number, program execution resumes at the command immediately following the end of the loop. The commands inside the loop are all indented to show that they are a part of the DO WHILE loop logic structure in order to improve readability.
- F. The CLOSE DATABASES command will de-activate (or close) all active database files.

- G. The final command, RETURN, terminates the program and returns control to the program that called it or to the normal dBASE III PLUS interactive mode (the dot prompt) if the user called the program directly.
- H. The ? commands by themselves have no effect but to space down a line and improve readability of the screen.

DO: SET THE ENVIRONMENT AND EXECUTE THE PROGRAM

Before executing the command file, be certain that you have the proper index file on your disk:

Type: .USE CUST <Return>

Type: .INDEX ON CUST TO CUSTNO <Return>

Now, be certain that CUST is backed-up before updating the file:

Type: .COPY TO CUST.BAK <Return>

To execute the command file:

Type: .DO UPDATE <Return>

DO: TEST THE PROGRAM

To test the program to see if it works enter a valid customer ID at the prompt:

Enter Customer ID, or 0 to Quit

Type: 1218 <Return>

The customer information for Richie Rich should be displayed on the screen. At the prompt:

Enter Amount Paid

Type: 0 <Return>

By entering a zero amount the update will have no effect on the various amount fields in this record.

Now, test the program to see if it ends properly:

Enter Customer ID, or 0 to Quit

Type: 0 <Return>

DO: MODIFY THE PROGRAM

If the program doesn't work, or works incorrectly, enter a zero customer ID to stop program execution or press <Esc>. Then edit the program:

Type: .MODIFY COMMAND UPDATE <Return>

DO: ENTER THE PAYMENT DATA

Be certain that the program is working correctly and then enter the following payment data (do not type dollar signs or commas in amount paid):

Customer No.	Name	Amount Paid
912	Same Saturate	\$5,700.00
3118	Freddy Famous	\$3,600.00
1356	Paul Powerful	\$1,600.00
3162	Arthur Nelson	\$1,200.00
2202	Doug Dealer	\$1,200.00
2201	David Derrick	\$1,800.00
2308	Guiseppi Sardini	\$2,400.00
1898	Belle Beauty	\$2,400.00
1285	Frank Financier	\$1,500.00
2341	Ronald Roller	\$5,400.00

DO: CREATE AND PRINT REPORTS

Prepare an Accounts Receivable Summary Report, ARSUMM, which shows the results of updating the customer accounts for the month's charges.

Type: .CREATE REPORT ARSUMM <Return>

Include columns for the customer number, customer name, amount paid, and amount owed. Print the report to the printer.

Type: .REPORT FORM ARSUMM TO PRINT <Return>

DO: LIST THE PROGRAM

Obtain a print (hard copy) of your program:

Type: .TYPE UPDATE.PRG TO PRINT <Return>

READ: Chapters 1 - 4 of Programming with dBASE III PLUS Reference Manual.

READ: A SUMMARY PROGRAM

The management of Al's Limousine Service are re-evaluating the lease prices, again, and want a report which shows the total for all lease revenue for the month, the average rental fee, and the total number of limousines rented.

DO: DEVELOPING A SUMMARY PROGRAM

If dBASE III PLUS is not active, you should start the program, set the default drive, and set the editing menus on. Since we are not going to be using the Assistant program for this unit, press <Esc> to leave Assist.

To enter a command file named SUMMARY in the dBASE III PLUS word processor:

Type: .MODIFY COMMAND SUMMARY <Return>

When dBASE Word Processor appears on the top line of the screen, begin typing in the program as shown below:

*program name: summary.prg author: joe programmer
*this program prints a summary of rental fees
*files used: auto.dbf

USE limo
SET DEVICE TO SCREEN
CLEAR
@ 3,15 SAY "PRINTING A SUMMARY OF RENTAL FEES AND AVERAGES"
SET DEVICE TO PRINT
@ 3,20 SAY "RENTAL FEES SUMMARY"

*perform calculations
SUM rental TO mtot_rent
AVERAGE rental TO mavg_rent
COUNT FOR cust>0 TO mcount

*print results
@ 6,15 SAY "Total Monthly Rental Fees:"
@ 6,42 SAY mtot_rent PICTURE "999,999.99"
@ 8,15 SAY "Average Monthly Rental Fee:"
@ 8,43 SAY mavg_rent PICTURE "9,999.99"
@ 10,15 SAY "Total Number of Limousines Rented:"
@ 10,51 SAY mcount PICTURE "999"

EJECT
SET DEVICE TO SCREEN
RELEASE ALL LIKE m*
CLOSE DATABASES
RETURN

To save the program in a command file:

Type: <Ctrl> W

READ: HOW THE PROGRAM WORKS

The following is a step-by-step explanation of how the program works, including an explanation of the new dBASE III PLUS keywords:

- A. The program is executed sequentially from top to bottom, line by line.
- B. The asterisks at the beginning of a line cause dBASE III PLUS to ignore that line so that it is not executed. These are comment lines only for the use of the programmer.
- C. The CLEAR command clears the screen and positions the cursor to the upper left-hand corner.
- D. The @ command is used to print (display) information in a specified format at a given set of coordinates (line,row). The SAY option is used to display constant information. The PICTURE option gives the user formatted output capability and enforces restrictions on the data that can be entered into a variable. Some formatting symbols that may be used in the PICTURE template are:
 - 9 - allows only digits for character data, digits and signs (+ or -) for numeric data
 - ! - converts letters to uppercase and has no effect on other characters
 - \$ - displays dollar signs in place of leading zeroes
 - * - displays asterisks instead of leading zeroes
 - . - specifies decimal position
 - , - comma only displayed if there are numbers to left of it
- E. The SET DEVICE command tells dBASE III PLUS that all succeeding @ commands will be routed to the screen or printer. (The default is the screen.)
- F. The SUM command totals all the monthly rental fees.
- G. The AVERAGE command calculates an average of all the monthly rental fees.
- H. The COUNT command counts the number of limousines that are currently rented.

- I. The EJECT command causes the printer to form feed.
- J. The RELEASE command deletes all memory variables that begin with the letter, M, and CLOSE DATABASES de-activates all active database files.
- K. The final command, RETURN, terminates the program and returns control to the program that called it or to the normal dBASE III PLUS interactive mode (the dot prompt) if the user called the program directly.

DO: EXECUTE, TEST, AND MODIFY THE PROGRAM

To execute the command file:

Type: .DO SUMMARY <Return>

If the program doesn't work, or works incorrectly, press <Esc>. Then edit the program:

Type: .MODIFY COMMAND SUMMARY <Return>

Notice when the program executes that dBASE III PLUS is providing the user with status information in reference to SUMming, AVERAGEing, and COUNTing. dBASE III PLUS provides this information for many of its commands. The programmer usually provides this status information to the user within the command file so that it is not necessary to have dBASE III PLUS repeat this information. SET TALK OFF will control what appears on the screen:

Type: .MODIFY COMMAND SUMMARY <Return>

Insert the line, SET TALK OFF, after the line, USE limo in the command file, save the program, and execute it. Notice the difference on the screen.

DO: LIST THE PROGRAM

Obtain a hard copy of the program:

Type: .TYPE SUMMARY.PRG TO PRINT <Return>

READ: Chapters 6 - 7 of Programming with dBASE III PLUS Reference Manual.

READ: REPORTS UNDER PROGRAM CONTROL

Many times it is not possible to create the desired report with the report generator. If the report involves more than one file, complex logical record selection, or special formatting, the report generator cannot be used.

If the report generator cannot be used, it will be necessary to write some type of dBASE III PLUS program to create the reports. Most reports can be produced satisfactorily using program control.

The general outline for programs which produce reports is as follows:

- (1) Index the file to the desired order before starting.
- (2) Initialize a page counter to 1 and the line counter to a high value. Set the totals accumulators to 0.

```
STORE 1 TO mpagectr
STORE 99 TO mlinectr
STORE 0.00 TO mttotal
```

- (3) If the report is to go to the printer, set the printer on and the console off as follows:

```
SET DEVICE TO PRINT
SET CONSOLE OFF
```

(SET CONSOLE OFF only after program has been tested and is working properly.)

- (4) Open the files and initiate the scan with a DO loop. SKIP records as necessary.
- (5) Before printing each line, check for line counter overflow and, if necessary, eject and print the heading. Increment page counter variable and reset line counter.

```
IF mlinectr>60
  EJECT
  STORE 6 TO mlinectr
  STORE mpagectr+1 TO mpagectr
ENDIF
```

- (6) Print the lines by using the @ command. Accumulate the totals.

```
@ mlinectr,1 SAY partno
@ mlinectr,10 SAY desc
@ mlinectr,40 SAY cost
STORE mtotal+cost TO mtotal
STORE mlinectr+1 TO mlinectr
```

- (7) Increment through the file with a SKIP command.
- (8) On completion, print the totals after the end of the loop.
- (9) During printing, tell the user that the printing is in progress.
- (10) Switch the console back on and the printer off upon completion.

READ: A REPORT PROGRAM

The management of Al's Limousine Service decide they need more detail concerning individual rental fees compared to the average rental fee amount. Since a report using the AVERAGE function would not be possible to create using the Report Generator function, you decide to develop the report under program control.

DO: DEVELOPING A REPORT PROGRAM

If dBASE III PLUS is not active, you should start the program, set the default drive, and set the editing menus on. Since we are not going to be using the Assistant program for this unit, press <Esc> to leave Assist.

To enter a command file named RENTAVG using the dBASE word processor:

Type: .MODIFY COMMAND RENTAVG <Return>

Type your command file so that the completed program looks like the following:

```

*program name: rentavg.prg      author: joe programmer
*this program prints an analysis of each rental fee compared to
*the average
*files used: work area 1-cust.dbf index custcust.ndx
*               work area 2-limo.dbf index limocust.ndx

```

```

*set processing environment
SET TALK OFF
SET DEVICE TO SCREEN
SELECT 1
USE cust INDEX custcust
SELECT 2
USE limo INDEX limocust
SELECT 1
SET RELATION TO cust INTO limo

```

```

*initialize variables
STORE 1 TO mpagctr
STORE 1 TO mlinectr
STORE .T. TO mnewpage
STORE 0 TO mtot_diff

```

```

CLEAR
@ 3,15 SAY "PRINTING AVERAGE RENTAL FEES ANALYSIS REPORT"
SET DEVICE TO PRINT
SELECT 2
AVERAGE rental TO mavg_rent
EJECT
SELECT 1

```

```

*scan the file one record at a time until end-of-file
DO WHILE .NOT. EOF()

```

```

    *print heading if new page
    IF mnewpage
        @ 3,20 SAY "ANALYSIS OF RENTAL FEES COMPARED TO AVERAGE"
        @ 5,1 SAY date()
        @ 5,73 SAY "Page"
        @ 5,78 SAY mpagctr PICTURE "99"
        @ 7,1 SAY "Customer Name"
        @ 7,31 SAY "Make"
        @ 7,54 SAY "Rent Fee"
        @ 7,64 SAY "Average"
        @ 7,72 SAY "High/Low"
        STORE 9 TO mlinectr
    ENDIF

```

```

    *print record if not deleted
    IF .NOT. DELETED()
        @ mlinectr,1 SAY TRIM(first)+" "+TRIM(last)
        @ mlinectr,31 SAY limo->make
        @ mlinectr,54 SAY limo->rental PICTURE "99999.99"
    ENDIF

```

WRITING SIMPLE PROGRAMS 12-14

```

        @ mlinectr,63 SAY mavg_rent PICTURE "99999.99"
        STORE limo->rental - mavg_rent TO mhigh_low
        @ mlinectr,72 SAY mhigh_low PICTURE "99999.99"
        STORE mtot_diff + mhigh_low TO mtot_diff
        STORE mlinectr + 1 TO mlinectr
    ENDIF

    *check for line counter overflow
    IF mlinectr>60
        EJECT
        STORE mpagectr+1 TO mpagectr
        STORE .T. TO mnewpage
    ELSE
        STORE .F. TO mnewpage
    ENDIF

    *move pointer to next record
    SKIP
ENDDO

*print total
STORE mlinectr + 1 TO mlinectr
@ mlinectr,31 SAY "TOTAL DIFFERENCE"
@ mlinectr,71 SAY mtot_diff PICTURE "999999.99"

EJECT

*restore original processing environment
SET DEVICE TO SCREEN
RELEASE ALL LIKE m*
CLOSE DATABASES
CLEAR
SET TALK ON
RETURN

```

To save the program in a command file:

Type: <Ctrl> W

READ: HOW THE PROGRAM WORKS

The following is a step-by-step explanation of how the program works, including an explanation of the new dBASE III PLUS keywords:

- A. The program is executed sequentially from top to bottom, line by line.
- B. The asterisks at the beginning of a line cause dBASE III PLUS to ignore that line so that it is not executed. These are comment lines only for the use of the programmer.

- C. The first section of the program sets the processing environment by setting processing parameters such as SET TALK OFF and by selecting work areas and using database files. Since linked database files are to be used in this program, a relation is also set in this section using the SET RELATION command.
- D. Any memory variables that are to be used globally in a program must be created and initialized with the STORE command before they can be used in a calculation or logical operation. The following global memory variables are used in this program:
- mpagectr - stores the current page number
 - mlinectr - stores the current line number
 - mnewpage - stores the new page condition for printing a heading at the top of the page
 - mtot_diff - stores the total differences between rent fee and average rent fee for all records
- E. The LIMO database is selected and the billed amount is averaged and stored to a memory variable. Then the CUST database is selected for processing.
- F. The DO WHILE begins the record processing loop. The loop continues to process each record of the database file while it is not the end of the file. When the end of the file is encountered, the EOF() function will become false and the loop will quit repeating.
- G. The IF/ENDIF is a programming structure that provides for conditional processing of commands. The structure begins with an IF and terminates with an ENDIF.

The first IF/ENDIF structure tests the condition of the memory variable, mnewpage. If it is true, a page heading is printed. If it is false, processing continues on the line immediately following the ENDIF.

The next IF/ENDIF structure tests to see if a record is marked for deletion. If it is marked for deletion, the record is not printed. If it is not marked for deletion, it is printed.

The next IF/ENDIF structure checks to see the current line to make sure it is not the end of a page. If the line counter variable is greater than 60, then a form feed (EJECT) is issued, the page counter variable is incremented by one, and the new page condition is set to true so that a heading will be printed.

The ELSE part of the IF/ENDIF structure is performed if the condition on the IF is false. In other words, if the line counter variable is not greater than 60, then a page heading does not need to be printed and so the new page variable is set to false.

- H. The SKIP moves the record pointer to the next record in the database file.
- I. The section of the program immediately following the ENDDO loop terminator print the total difference for all records between the rental fee and the average fee.
- J. The EJECT sends the printer a form feed to conclude all printing.
- K. After all processing is complete, the processing environment should be returned to the state to which it was when processing began.

DO: SET THE ENVIRONMENT AND EXECUTE THE PROGRAM

Before executing the command file, be certain that you have the proper index files on your disk:

Type: .USE CUST <Return>

Type: .INDEX ON CUST TO CUSTCUST <Return>

Type: .USE LIMO <Return>

Type: .INDEX ON CUST TO LIMOCUST <Return>

Now, execute the command file:

Type: .DO RENTAVG <Return>

DO: TEST AND MODIFY THE PROGRAM

If the program doesn't work, or works incorrectly, press <Esc>. Then edit the program:

Type: .MODIFY COMMAND RENTAVG <Return>

DO: LIST THE PROGRAM

Obtain a hard copy of the program:

Type: .TYPE RENTAVG.PRG TO PRINT <Return>

READ: Chapters 11 - 15 of Programming with dBASE III PLUS Reference Manual.

DO: DBASE III PLUS POWER-OFF PROCEDURE

To stop dBASE III PLUS processing:

Type: .QUIT <Return>

READ: dBASE III PLUS OPERATIONAL RULES

Having learned many dBASE III PLUS commands, it would be advisable at this point to establish some guidelines for using the dBASE III PLUS command language interactively as well as for command files. These are some rules that should be followed to insure that commands are properly written:

1. Each command must begin with a command verb.
2. Each command must conform to the syntax for that command.
3. A command consists of a verb and optionally one or more clauses that qualify the verb. Optional clauses (for instance, those beginning with FOR, WHILE, and NEXT) may occur in any order. For example, the following are equivalent.

```
.DISPLAY NEXT 25 FOR Name = 'Jones'  
.DISPLAY FOR Name = 'Jones' NEXT 25
```

4. The maximum command length is 254 characters.
5. Words within a command may be separated by any number of blank spaces. Blank spaces, however, are counted in the 254 character limit.
6. Commands and certain keywords may be abbreviated to the first four characters (for example, DISPLAY MEMORY can be abbreviated as DISP MEMO).
7. Command verbs, keywords, field names, memory variable names, and file names may be in uppercase, lowercase, or any mixture.
8. Although dBASE III PLUS has no prohibited words, it is advisable to avoid using keywords for file names, field names, or memory variables as they may cause some difficulty. For example, a field named Status could not be displayed by the command DISPLAY STATUS because the command would display the current processing status rather than the content of the field Status.

9. When programming, care must be taken that all DO WHILE/END DO, DO CASE/ENDCASE and IF/ENDIF statements fit entirely within one another. If for example, an IF statement is used within a DO loop, both the IF and its accompanying ENDIF must be contained between the DO WHILE and the ENDDO.

```
DO WHILE .NOT. EOF()  
  IF TRUE  
    ? 'HOORAY'  
  ENDIF  
ENDDO
```

EXERCISES

1. DEVELOPING AN UPDATE PROGRAM

Using the COLL database from a previous application, develop a program, DISPCOLL, which displays the student information on the screen for a selected student. (This program could serve as a skeleton for an update program by adding INPUT statements for each field you want to edit or update.)

The program should ask for the student's social security number to be input, find the student in the file, and display the field information on the screen.

For Example: DO WHILE mssno > ""

Don't use the DISPLAY command to display the field information, but use the @..SAY..PICTURE to format field contents output to the screen. To stop processing, have the program prompt the user to just press <Return> (don't enter a social security no.).

Obtain a list of the program (.TYPE DISPCOLL.PRG TO PRINT).

2. DEVELOPING A SUMMARY PROGRAM

Using the HIGH and COLL databases from a previous application, develop a summary program, GRDSUMM, develop a program which will print the average high school GPA and average SAT score for all students in the HIGH database file as well as the total number of high school students in the file.

Also include on the same report the average college GPA for all students in the COLL database and the total number of college students in the file.

Obtain a list of the program (.TYPE GRDSUMM TO PRINT).

3. DEVELOPING A REPORT PROGRAM

Using the COLL database from a previous application, develop a report program, GRDMAJOR, which prints the student name, social security number, total hours, grade points, and major for a selected major.

The report program should prompt the user to enter a major and then print the student information for the selected major.

For example: INPUT "Enter a Major " TO mmajor

 IF .NOT. DELETED()
 IF TRIM(major) <> mmajor

 ENDIF
 ENDIF

Obtain a list of the program (.TYPE GRDMAJOR TO PRINT).

LESSON 13

SYSTEM DESIGN AND PROGRAM DEVELOPMENT

OBJECTIVES:

- * Explain system design and analysis
- * Explain structured design and programming
- * Design a system using dBASE III PLUS
- * Develop a main menu program
- * List appropriate program development procedures using dBASE III PLUS

REQUIRED MATERIALS:

- * Paper and pen or pencil
- * dBASE III PLUS System Diskettes 1 and 2
- * A practice diskette
- * IBM PC/XT, PS/II, or compatible with 2 disk drives or 1 disk drive and hard disk, and printer

READ: SYSTEM DESIGN AND ANALYSIS PROCESS

The design of the database file is one of the early steps in the design of a system. Normally, several files are used as part of a total system. Various programs are used to process these files and to create a variety of reports. The design flow can easily be seen in the following outline.

I. Analysis Phase

1. Definition of the system objectives
2. Definition of the output information needed to meet these objectives.
3. Definition of output reports, forms, and displays to meet these objectives. These are defined as output specifications.
4. Definitions of input information needed to obtain the output forms, reports, and displays. These are defined as input specifications.

II. Design Phase

1. The functions of the system are separated into clearly defined functional entities as seen by the user.
2. The files are designed to support the specific functions.
3. The indexes for each file are defined.
4. A collection of programs is defined to support each function.

III. Programming

1. The main menu program is written that defines each of the functional modules.
2. Submenus are created for each function.
3. The programs are written in a modular method, implementing one module (or function) at a time.

READ: STRUCTURED DESIGN AND PROGRAMMING

This approach is more formally known as design analysis and structured design. Although the definition of structured design varies with various writers, certain key components of the definition are well-defined. The end product of the structured design approach is a product that is easy to read, understand, and update as necessary. This involves two steps:

1. Envisioning a logical structure for the system and programs.
2. Using a top-down approach in the design, defining the large-scale functions first and gradually working toward smaller components.

Some additional features of structured design and programming using dBASE III PLUS are:

- A. Top-down design approach. Begin by defining goals, inputs, and outputs.
- B. A modular design approach should be used. Design from the top down.
- C. Each module has a single entry and single exit.
- D. Each module performs a single function.
- E. Each module should be documented with name, function, input, and output requirements, other module requirements, last update, and any special notes needed.

- F. Each module should be limited to about 50 lines of code. Use comments liberally.
- G. Logic controls permitted include IF/THEN/ELSE, DO/WHILE, DO/UNTIL, PROCEDURE, AND CASE.
- H. Indentation should be used after all logic controls.

dBASE III PLUS forces the user to think in terms of structured design and uses a structured programming language. With dBASE III PLUS all programs are modular and often small modules are used in many programs. Programs are easy to read and update.

READ: DESIGNING A SYSTEM USING DBASE III PLUS

The management of Al's Limousine Service has decided that having each employee learn to use the various dBASE III PLUS commands in order to perform all the required processing steps is much too complex and time-consuming. They have decided to implement a complete system of processing using dBASE III PLUS command files.

You have identified the following processing functions:

- (1) Add/Edit Limousine Information
- (2) Update Monthly Rentals to Customer File
- (3) Add/Edit Customer Information
- (4) Record Customer Payments
- (5) Print Automobile Master List
- (6) Print Automobile by Customer Register
- (7) Print Customer Master List
- (8) Print Customer Statements
- (9) Print Accounts Receivable Report
- (10) Record Company Information
- (11) Pack and Reindex Databases
- (12) Backup and Restore Databases

Next, you determined that all of these processing functions could fit into the following categories:

- A. Process Limousine Information
- B. Process Customer Information
- C. Record Company Information
- D. File Maintenance

DO: FILE STRUCTURE PLANNING CHARTS

- 1. Based on these processing functions, complete a File Structure Planning Chart for each database. The following is the information for the LIMO.DBF database:

DATABASE NAME: LIMO.DBF
DESCRIPTION: STORES RENTAL INFORMATION ABOUT EACH LIMOUSINE
INDEX FILES: LIMOLIC.NDX -- INDEXED BY LICENSE
 LIMOCUST.NDX -- INDEXED BY CUSTOMER ID
REPORT FILES: LIMOMAST.FRM -- PRINTS LIMOUSINE MASTER LIST
 LIMOCUST.FRM -- PRINTS LIMOUSINE CUSTOMER REGISTER

FIELD NAME	TYPE	WIDTH	DECIMALS	PURPOSE/FUNCTION
LIC	C	7		License No.
MAKE	C	20		Make of Limousine
YEAR	N	4		Year Manufactured
CUST	N	4		Customer ID
Rental	N	7	2	Monthly Rental Fee

- Now complete a File Structure Planning Chart for the CUST.DBF database based on these processing functions and information presented in previous modules.

DO: STRUCTURE CHARTS

A Structure of Hierarchy Chart illustrates the system design. Each processing function is called a module and is shown as a box on the Structure Chart.

In this modular design approach, called top-down design, a higher-level module may flow to a lower-level module and a lower-level module always returns to the higher-level module that calls it.

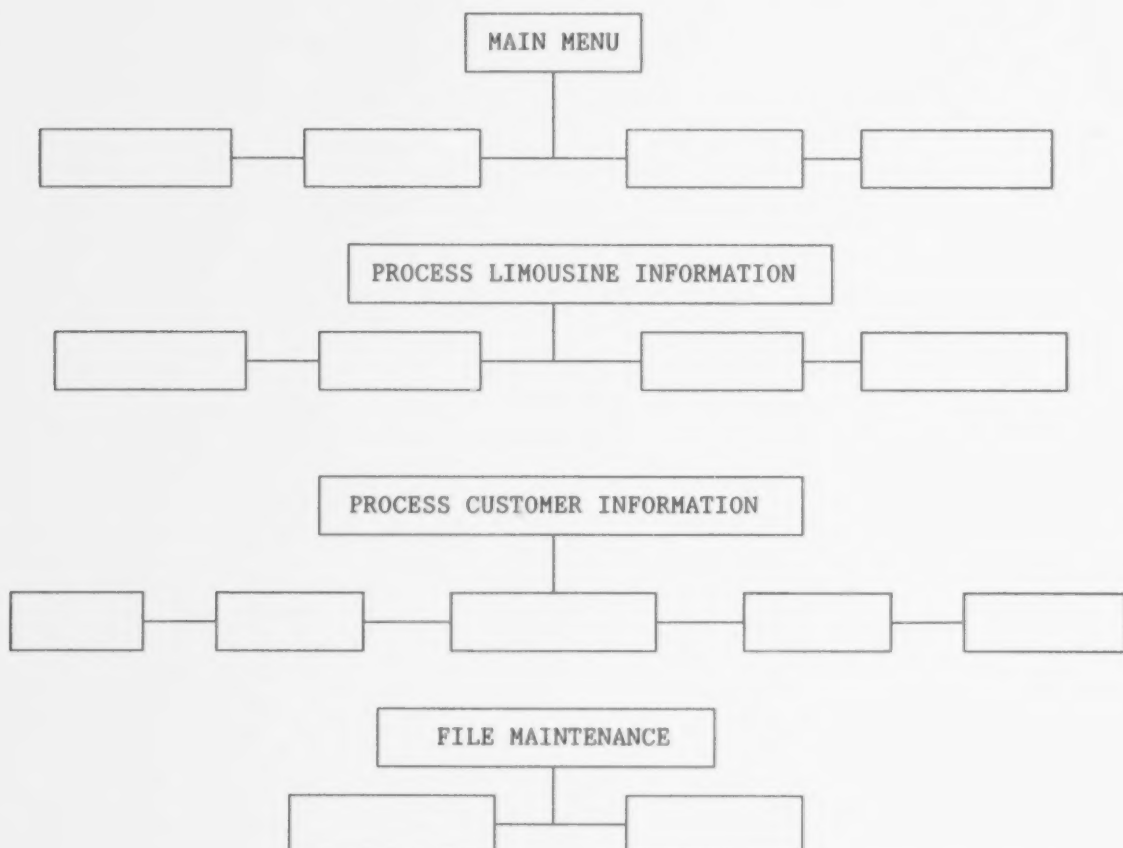
Using the following Structure Chart Form, complete the system design for Al's Limousine Service system based on the identified processing functions:

- A. Process Limousine Information
 - (1) Add/Edit Limousine Information
 - (2) Update Monthly Rentals to Customer File
 - (3) Print Limousine Master List
 - (4) Print Limousine by Customer Register
- B. Process Customer Information
 - (1) Add/Edit Customer Information
 - (2) Record Customer Information
 - (3) Print Customer Master List
 - (4) Print Customer Statements
 - (5) Print Accounts Receivable Report
- C. Record Company Information

- (1) Pack and Reindex Databases
- (2) Backup and Restore Databases

STRUCTURE CHART FORM

DESCRIPTION: _____



DO: DEVELOPING A MAIN MENU PROGRAM

Al's Limousine Service system should have a Main Menu Program. This program will display the main processing areas and allow the user to easily access these areas by making a choice from a menu. This menu program would advance to other menu programs which display the processing functions which in turn would advance to programs which perform the functions.

If dBASE III PLUS is not active, you should start the program, set the default drive, and set the editing menus on. Since we are not going to be using the Assistant program for this unit, press <ESC> to leave Assist.

To create a Main Menu program, LIMOMENO:

Type: .MODIFY COMMAND LIMOMENO <Return>

When dBASE Word Processor appears on the top line of the screen, begin typing in the program as shown below:

```
* program name: limomeno.prg      author: joe programmer
* this is the main menu program for the limo system
```

```
* set processing environment
SET DEVICE TO SCREEN
SET TALK OFF
CLEAR
```

```
*main control loop
DO WHILE .T.
```

```
    STORE " " TO inselect
    STORE .T. TO clr_screen
```

```
    *test for clearing screen
    IF clr_screen
        CLEAR
```

```
    ENDIF
```

```
    *display menu options on screen
```

```
    @ 1,20 SAY "*****"
    @ 2,20 SAY "                    AL'S LIMOUSINE SERRVICE                    "
    @ 3,20 SAY "*****MAIN MENU*****"
    @ 4,31 date ( )
    @ 6,20 SAY "1 - Process Limousine Information"
    @ 7,20 SAY "2 - Process Customer Information"
    @ 8,20 SAY "3 - Record Company Information"
    @ 9,20 SAY "4 - File Maintenance Activities"
    @ 10,20 SAY "5 - Stop Processing"
    @ 14,20 SAY "Enter your choice --->"
    @ 14,44 GET inselect PICTURE "9"
    READ
```

```

STORE .T. TO clr_screen

DO CASE
  CASE inselect = "1"
    DO limomen1
    *advance to limousine processing

  CASE inselect = "2"
    DO limomen2
    *advance to customer processing

  CASE inselect = "3"
    DO limomen3
    *advance to record company information

  CASE inselect = "4"
    DO limomen4
    *advance to file maintenance activities

  CASE inselect = "5"
    CLEAR
    QUIT
    *advance to operating system

  OTHERWISE
    WAIT "<INVALID CHOICE-PRESS ANY KEY>" TO mwait
    STORE .F. TO clr_screen

ENDCASE
ENDDO

```

To save the program in a command file:

Type: <Ctrl> W

READ: HOW THE PROGRAM WORKS

The following is a step-by-step explanation of how the program works, including an explanation of the new dBASE III PLUS keywords:

- A. The program is executed sequentially from top to bottom, line by line.
- B. The asterisks at the beginning of a line cause dBASE III PLUS to ignore that line so that it is not executed. These are comment lines only for the programmer's use.
- C. The first section of the program set the processing environment by setting processing parameters such as SET TALK OFF.

D. A main control loop is set up as a never-ending, or infinite, loop. DO WHILE .T. is always a true statement and will cause the loop to continue endlessly. This is a valuable technique to use in a menu program.

E. The memory variable used inside the loop are initialized:

```
inselect    -   stores the user's menu selection
clr_screen  -   stores the condition for clearing the
                 screen or not
```

F. The main menu is displayed on the screen. The GET is an optional statement used with the @ command to display variable information that may be either an existing memory variable or any field in the active database. This portion of the command does not invoke the full-screen mode, but must be activated by a READ command. The READ command activates all @ ...GETs and is most commonly used in command files for full-screen entry or editing of data. At the READ command the program halts execution until the data is input to all active GETs--in this case the one prompting for the variable, inselect.

G. The DO CASE is a structured programming command that selects only one course of action from among a set of alternatives. The DO CASE/ENDCASE is like a series of IF/ENDIF structures. The DO CASE must be terminated with ENDCASE. The first CASE condition that is true is selected and executed. After on, CASE is found and its associated commands are processed, no further CASE statements are evaluated. The OTHERWISE is executed if all previous CASEs are evaluated as false.

H. Each CASE statement is followed by a DO statement that executes another dBASE III PLUS command file. The Main Menu command file is used to execute other command files. When the RETURN statement is executed in the called command file, it returns to the command file which called it. Thus, top-down design is quite effective using dBASE III PLUS.

DO: EXECUTE, TEST, AND MODIFY THE PROGRAM

To execute the command file:

```
Type:      .DO LIMOMENO  <Return>
```

Any choice that is selected at this point will result in a file not found error except choice 5 because the other programs have not been developed yet.

If the program doesn't work, or works incorrectly, press <ESC> and terminate execution of the command file.

Then edit the program:

Type: .MODIFY COMMAND LIMOMENO <Return>

DO: LIST THE PROGRAM

Obtain a hard copy of the program:

Type: .TYPE LIMOMENO.PRG TO PRINT <Return>

READ: PROGRAM DEVELOPMENT PROCEDURES USING dBASE III PLUS

The following steps are recommended to ensure that programs are readable by both you and users who will work with them. You should incorporate all these procedures in every program you write.

1. All programs should have comments at the beginning that contain the following information:

Name of program and function description
Copyright notice if applicable
Author
File name
Input files and index used
Additional files used
Version, date, programmer, initials.

For example:

* program name: invupd.prg - author: Joe Programmer
* updates inventory transactions to master
* files: invtrans.dbf, invmast.dbf, inv_no.ndx
* date updated: 02/13/83, 03/10/83, 06/20/83

2. Programming structure should conform to the following:

- a. All dBASE reserved words should be capitalized
- b. All words not reserved should be lowercase
- c. IF and DO loop closures should be commented upon
- d. Command lines should be less than 100 characters
- e. IF and DO commands are indented consistently

3. Comments should be used liberally throughout the program to explain what is happening. This will not greatly affect programming speed, but to keep programs as fast as possible, use comments outside a loop to which they refer.

For example:

```
* update inventory on part no.  
DO WHILE .NOT. EOF
```

4. DO calls to additional programs will slow down the main program somewhat since the additional program file must be opened. As a general rule, the file is opened only the first time the program is called. If multiple calls to another program or procedure are used, it is just as fast to use the DO call to the additional program as to incorporate the program code in the mainline program.
5. All programs and file names should conform to a file naming convention that is documented for your application.
6. Programs using the screen display should display a header showing the company name, functions, and date. The header should always remain on the screen unless it is replaced by another header. For example:

CLEAR

```
@ 1,1 SAY '*****WIDGET MANUFACTURING*****'  
@ 2,1 SAY 'INVENTORY ADD'  
@ 2,60 SAY DATE ( )
```

7. Do not use dBASE III PLUS reserved words as variable names.
8. If the program must spend more than 5 seconds printing, indexing, or sorting, it should tell the user what is going on, possibly, display a changing message. For example, a program updating a mailing list file should display the record number or name it is currently processing.
9. Use as many help messages as possible to guide the user. Don't assume the user has a manual or instructions. For instance, a report program should request the user to load the paper.
10. If a program discovers an error in a file, it should display the cause of the error to the user in plain English and give sufficient information to aid in tracking the error. Keep the program running if at all possible.
11. If a multiple branch is needed in a program (as with a menu program), use a DO CASE command instead of an IF command. The DO CASE command is faster.
12. Use menus to control the access and operation of programs. Divide programs into functional groups and use submenus as necessary for each group. Don't expect the user to remember program names.

13. Be sure systems include programs for data file recovery and backups, and a program to re-index all files.
14. In starting a program that should not be interrupted (such as a posting program), be sure to set ESCAPE OFF and then, when the program is finished set ESCAPE ON.
15. Reserve the twenty-third or twenty-fourth line of the screen for error messages.
16. Add delays and ring the terminal bell when displaying error messages to get the user's attention. Allow the user time to read the message.
17. Always close files that are not being used. An easy way to do this is to use a CLOSE DATABASE command at the end of programs which open a database file or files.
18. Release variables as often as possible using the RELEASE command. If variables must cross a submenu boundary (such as company name), save these in a .MEM file.
19. Before writing a program, do the necessary design analysis, data flow diagrams, and file designs. This will minimize the errors and rewriting of the final programs.
20. Input from the user should be interpreted uppercase in key fields so that the user can enter the selection in either upper- or lower-case. Use the ! function:

```
CASE UPPER(option) = "A"  
DO invadd
```

READ: dBASE III PLUS STANDARD PROGRAM FORMAT

A standard format for your dBASE III PLUS programs should be developed and followed. This will make writing a dBASE III PLUS program a more natural and efficient procedure.

The following pseudocode presents a format which will fit most dBASE III PLUS programs including menus, screen input programs, and print programs. This should be used until one better suited to your own programming style may be developed.

*Comments including program name, description, and author

Set input/output format and program environment

Use database files if needed

Begin a loop, if one is needed

 Present screen or menu of options
 and/or

 Get user input or read data from database file
 and/or

 Process the input, perform calculations, or print data

Loop back again until finished

Release memory variables

Restore program environment to initial settings

EXERCISES

1. OSCAR'S OFFICE MACHINES

This exercise is to design a hypothetical business system and develop a Main Menu program to manage the business applications of the system:

Oscar's Office Machines is in the business of selling and servicing office machines such as typewriters, calculators, and copiers. Each time Oscar sells an office machine, he tries to sell a service contract with the machine to guarantee all parts and labor for the machine during the period that the service contract is in effect. If the machine breaks down during the contract period, the customer does not have to pay for the repairs to the machine if there is a service contract for the machine.

Service contracts are renewed annually. However, each are renewed on different dates since the contracts were sold on different dates. Oscar's problem is keeping track of service contracts that need to be renewed. He would like to be able to keep a list of contracts that are going to expire in a certain month so that he can send a new contract to the customer before it expires.

Oscar has purchased a microcomputer and dBASE III PLUS to help maintain his service contract system. However, he has decided that developing the service contract system is much too complicated for him to take time to learn, so he has hired you to develop this system for him.

You have determined from your interviews with Oscar that each service contract may have one or several office machines included since many customers buy more than one office machine from Oscar. This being the case, you have designed the following database file specifications:

CUSTOMER CONTRACT FILE

Contract No. (C)
Contract No. (C)
Contract Name (C)
Contract Address (C)
Contract City (C)
Contract ZIP (C)
Expiration Date (D)
Contract Total (N)

CONTRACT DETAIL FILE

Contract No. (C)
Serial No. (C)
Model (C)
Kind (TYPE, CALC, COPY) (C)
Amount (N)

INDEX = Contract No.

INDEX = Contract No. + Serial No.

You have determined that the following system functions are needed:

- (1) Update Customer Contracts
 - a. Add Customers
 - b. Edit Customers
 - c. Review Customers
 - d. Delete Customers
- (2) Update Contract Detail
 - a. Add Detail
 - b. Edit Detail
 - c. Review Detail
 - d. Delete Detail
- (3) Process Contract Totals
- (4) Print Reports
 - a. Print Contract Master List
 - b. Print Contract Expiration Schedule
 - c. Print Service Contracts

You have decided that dBASE III PLUS is most effective using a modular system comprised of several small, simple programs to address each business function. Each module will be a separate program that does one part of the total job. Through this modular approach you can build the system one program at a time with the advantage of being able to change or add a program with relative ease. To tie all of the individual programs together into one coherent package, you will use a menu system. A Main Menu will advance to submenus for each of the primary business functions.

(1) DEVELOP A STRUCTURE CHART

Using the structure chart forms, develop a structure chart of our business system which shows each business function as a module and a corresponding menu name for each module in our system.

(2) Create file structure planning charts for the Customer and Contract Detail database file.

(3) CREATE THE MAIN MENU PROGRAM

Develop a Main Menu program that will select a submenu for each of the primary functions in your system design. The Main Menu (and all menus, for that matter) should display the name of the business as well as the current date in addition to the choices that can be made from the menu.

The Main Menu should be able to advance to each of the submenus. All submenus return to the Main Menu. All programs from the submenus return to the submenu from which they were called. Test the program and obtain a print.

LESSON 14

APPLICATIONS GENERATOR

OBJECTIVES:

- * Explain the features of the Applications Generator
- * How to start the Applications Generator and select options
- * Changing the color display
- * Generating and running an application
- * Creating a custom application

REQUIRED MATERIALS:

- * Paper and pen or pencil
- * dBASE III PLUS System Diskettes 1 and 2
- * A practice diskette with the Applications Generator program
- * A practice diskette
- * IBM PC/XT, PS/II, or compatible with 2 disk drives or 1 disk drive and hard disk, and printer

READ: WHAT THE APPLICATIONS GENERATOR DOES

You may wonder what the Applications Generator is. Before we discuss what it is, let's consider what it does. If you're too busy to write programs or otherwise not so inclined to write dBASE programs, then you might want to use the Applications Generator. The Applications Generator will help you to:

- o Learn about dBASE III PLUS without having to work through the manuals
- o Create database files and indexes
- o Set up and use simple menu-driven applications without knowing or having to use dBASE programming
- o Design customer screen entry forms to enter and change data easily
- o Create reports and generate mailing labels

You have already learned to perform all these operations. However, in addition to these functions, the Applications Generator can help you to:

- o Design custom applications
- o Modify existing applications, including those created by the Applications Generator
- o Learn the basics of dBASE programming by reviewing the program code generated by the Applications Generator

READ: WHAT THE APPLICATIONS GENERATOR IS

The Applications Generator is a program written in the dBASE programming language to create other dBASE programs. Although the Applications Generator is not a substitute for dBASE III PLUS, it is a shortcut to get you started or to reduce development time.

However, the Applications Generator cannot do everything for you. When developing sophisticated systems and/or applications, you may have to know more about programming than the Applications Generator is able to handle.

DO: STARTING THE APPLICATIONS GENERATOR

If dBASE III PLUS is not active, you should start the program, set the default drive, and set the editing menus on. Since we are not going to be using the Assistant program for this unit, press <Esc> to leave Assist.

1. If you are not using a hard disk, place the disk with the Applications Generator on it in Drive B:

Type: .SET DEFAULT TO B <Return>

2. For hard disk or floppy disk, to begin the Applications Generator:

Type: .DO APPSGEN <Return>

The Applications Generator main menu appears on the screen:

dBASE III PLUS
APPLICATION GENERATOR MENU

1. CREATE DATABASE
2. CREATE SCREEN FORM
3. CREATE REPORT FORM
4. CREATE LABEL FORM
5. SET APPLICATION COLOR
6. AUTOMATIC APPLICATION GENERATOR
7. RUN APPLICATION
8. ADVANCED APPLICATION GENERATOR
9. MODIFY APPLICATION CODE

0. EXIT

select █

Command ||<C:>|| || || ||

READ: MAKING SELECTIONS AND EXITING

To select a menu option, type the appropriate number key then press the <Return> key to activate the selection. The default selection is 0 to exit. By pressing <Return> without selecting a menu option will return you to the dBASE III PLUS dot prompt.

If you type the wrong selection, just type the correction before pressing the <Return> key. If you activate a selection by mistake, press the <Return> key to back out of the operation and return to the Applications Generator main menu.

READ: THE MENU CHOICES

All menu options do not have to be used. In the following exercises we will not work with options 1-4 since these are all operations you have performed in previous units. In fact these operations will work exactly the same way as you have already learned. So we will use database files, indexes, screen forms, reports, and labels that you have already developed to generate an application.

DO: CHANGING THE COLOR DISPLAY

If you have a monochrome display, then skip this section. The Applications Generator uses the default color settings of your video display. If you want different color settings for an application:

1. To change the color settings, at the Applications Generator main menu, type 5 and press <Return>.
2. Follow the coding chart at the top of the screen to set the foreground and background colors for standard text, enhanced text, adding and editing information, and the border.
3. When you finish, press <Ctrl> <End> and type S to save your changes. If at some later point you decide you do not like the colors you have selected, you may perform this operation again to change the color settings.

DO: THE AUTOMATIC APPLICATION GENERATOR

We are going to create an application for the LIMO database using the database file, indexes, screen format, reports, and labels that you previously developed. Then you will repeat the process for the CUST database.

The application will provide a menu of what operations it can do. You will be able to add information, change information, remove information, and view information, merely by choosing the correct menu selection:

1. To create an application for the LIMO database, at the Applications Generator main menu type 6 and press <Return>.

Notice the help message at the top of the screen. For each step in creating an application, there's a different help screen. Refer to these messages for explanatory information about the current option.

2. When you are prompted to, Enter APPLICATION AUTHOR:, type your first and last name and press <Return>.

It will put your name in the program coding as the author of this application. It also supplies today's date, provided you typed it in when you powered on the computer.

3. When you are prompted to, Enter APPLICATION FILE NAME:, type LIMO and press <Return>.

This is the name of the command or program file which will contain the dBASE coding for your application.

4. When you are prompted to, Enter DATABASE FILE NAME:, type LIMO and press <Return>.

5. When you are prompted to, Enter INDEX FILE NAME:, type LIMOLIC. If an index file with this name is already set up, the Applications Generator uses it. Since no index file exists with the name, the Applications Generator creates one for you.

It requests the key field for the index file. Remember that the key field is the field on which the database is to be ordered. In this example the key field is the LICENSE field.

When you are prompted to, Enter KEY FIELD NAME:, type LICENSE and press <Return>. The Applications Generator checks that the field exists and, if so, tells you that it is creating the index file.

6. When you are prompted to, Enter SCREEN FORMAT NAME:, type LIMOFORM.
7. When you are prompted to, Enter REPORT FORM NAME:, press <Return> to bypass this option since we don't have a report form to use.
8. When you are prompted to, Enter LABEL FORM NAME:, press <Return> to bypass this option since we don't have a label form to use.
9. When you are prompted to, Enter APPLICATION MENU HEADING:, type LIMOUSINE DATABASE MAIN MENU and press <Return>.

Now that you have supplied the necessary information, the Application Generator writes the dBASE program code for the application. You see this code scroll by on the screen. You don't have to understand the code to use your application. When the code generation is finished, you are returned to the Application Generator main menu. The new application is now stored in a program file named, LIMO.PRG. The Application Generator also created a memory file, LIMO.MEM to store memory variables used with the application. You need both of these files to run the application.

DO: RUNNING YOUR APPLICATION

1. To run your application, at the Applications Generator, main menu type 7 and press <Return>.
2. When you are prompted to, Enter the Application Name:, type LIMO and press <Return>.

Your application's menu is now displayed on the screen:

LIMOUSINE DATABASE MAIN MENU

1. ADD INFORMATION
2. CHANGE INFORMATION
3. REMOVE INFORMATION
4. REVIEW INFORMATION
0. EXIT

select █

Command █<C:>█LIMO

█Rec:20/20 █ █

Now let's try some of the choices.

3. Use option 1, **ADD INFORMATION**, to add the following limousine data:

ALS-99, Cadillac, 1987, \$1500.00, 0

Try options 2 and 4 to edit and/or review the data in your database. You can delete records while changing or reviewing by pressing <Ctrl> U when you see the record that you want deleted.

These options correspond with the following dBASE commands:

- | | | |
|-----------------------|--------|---------|
| 1. ADD INFORMATION | -----> | .APPEND |
| 2. CHANGE INFORMATION | -----> | .EDIT |
| 3. REMOVE INFORMATION | -----> | .PACK |
| 4. REVIEW INFORMATION | -----> | .BROWSE |
| 0. EXIT | -----> | .RETURN |

Press 0 to exit and return to the Applications Generator main menu.

DO: GENERATE AND RUN ANOTHER APPLICATION

Now repeat the previous two operations for the CUST database using the following information:

Application Author:	Your Name
Application File Name:	CUST
Database File Name:	CUST
Index File Name:	CUSTID
Screen Format Name:	CUSTFORM
Report Format Name:	CUSTRPT1
Label Format Name:	CUSTLBL1
Application Menu Heading:	CUSTOMER DATABASE MAIN MENU

APPLICATIONS GENERATOR 14-6

Run the CUST application and use menu to PRINT REPORT and PRINT LABELS. Then press 0 to exit and return to the Applications Generator main menu.

DO: CREATING A CUSTOM APPLICATION

The Advanced Applications Generator option will let you generate custom applications that issue standard dBASE III PLUS commands. The Advanced Applications Generator will allow you to determine what actions your application is to perform. You can also modify an existing application using this option.

We are going to create another menu program that will join the LIMO and CUST applications together to form a complete dBASE processing system:

1. To create a custom application, at the Applications Generator main menu type 8 and press <Return>.
2. When you are prompted to, Enter APPLICATION AUTHOR:, type your first and last name and press <Return>.
3. When you are prompted to, Enter APPLICATION FILE NAME:, type ALSMAIN and press <Return>.
4. When you are prompted to, Enter DATABASE FILE NAME:, press <Return> to bypass this option.
5. When you are prompted to, Enter INDEX FILE NAME:, press <Return> to bypass this option.
6. When you are prompted to, Enter SCREEN FORMAT NAME:, press <Return> to bypass this option.
7. When you are prompted to, Enter REPORT FORM NAME:, press <Return> to bypass this option.
8. When you are prompted to, Enter LABEL FORM NAME:, press <Return> to bypass this option.
9. When you are prompted to, Enter APPLICATION MENU HEADING:, type AL'S LIMOUSINE SERVICE and press <Return>.
10. The Applications Generator presents a setup screen for your menu. You can set up as many as nine menu choices. Choice 0 is by default established as the Exit option, just as in the other menus.

If you type Y for yes, the Applications Generator asks if you want to edit the existing menu entries. You can delete a menu item by pressing <Ctrl> Y at the beginning of a setup line.

READ: MODIFY APPLICATION CODE

Option 9, MODIFY APPLICATION CODE, runs the dBASE Word Processor with a supplied program file name. This is primarily a convenience for dBASE programmers who are working with the Applications Generator.

DO: PRINT PROGRAM LISTINGS

You may obtain a print of the applications you have created in this unit by using the TYPE command. Select option 0 from the Application Generator main menu to return to the dBASE III PLUS dot prompt.

To print the LIMO application:

Type: .TYPE LIMO.PRG TO PRINT <Return>

To print the CUST application:

Type: .TYPE CUST.PRG TO PRINT <Return>

To print the ALSMAIN applications:

Type: .TYPE ALSMAIN.PRG TO PRINT <Return>

DO: DBASE III PLUS POWER-OFF PROCEDURE

To stop dBASE III PLUS processing:

Type: .QUIT <Return>

EXERCISES

1. OSCAR'S OFFICE MACHINES

Using the Applications Generator, perform the following options to create an application for Oscar's Office Machines service contract system (All these operations may be performed from the Applications Generator main menu):

- (1) Create a database using the file specifications from the previous unit. Index as specified.
- (2) Create a screen forms for both databases.
- (3) Create a Customer Contract List report that prints all the information from the Customer Contracts database file about each contract customer.
- (4) Create customer name and address labels.
- (5) Generate an application for each database file according to your Structure Chart Forms.
- (6) Run the applications and add the following service contract data, print the report, and print labels.
- (7) Obtain a print of your applications by using the TYPE command.

Use the following service contract data:

C002, Anderson Abstract Company, 1200 SW 29th, Oklahoma City, OK, 73159, 2/12/87:

612171556	IBM Selectric III	TYPE	\$75.00
61268954	IBM Selectric III	TYPE	\$75.00

C001, Associated Travel, 500 S. Brookhaven, Oklahoma City, OK, 73119, 1/15/87:

0756678	Minolta 3000	COPY	\$125.00
---------	--------------	------	----------

C003, Brickland and Brimer, Attys., 212 W. 7th, Moore, OK, 73200, 3/5/87:

1467372	Remington Rand	TYPE	\$60.00
1212387-A	Sharp 300C	CALC	\$30.00

C004, Calvary Baptist Church, 700 S. Kentucky, Oklahoma City, OK, 73159, 3/30/87:

6981982	IBM Quietwriter II	TYPE	\$85.00
1340537	IBM Quietwriter III	TYPE	\$95.00
26-6467457	Panasonic 300D	COPY	\$150.00

C005, Citizens National Bank, 100 N. Broadway, Oklahoma City, OK, 73135, 4/2/87:

5383946	IBM Selectric III	TYPE	\$75.00
5384027	IBM Selectric III	TYPE	\$75.00
9111904	Facit 1850	TYPE	\$65.00
0454728	Facit 1950	TYPE	\$75.00

C007, Doctor's Clinic, 105 SW 2nd, Norman, OK, 73134, 4/25/87:

6164097	Olympia 45	TYPE	\$55.00
0173951	Olympia ET 121	TYPE	\$65.00

C009, Graves Oil Company, 1000 SW Expressway, Midwest City, OK, 73245, 5/22/87:

2146C	Sharp 300C	CALC	\$30.00
7013389	IBM Selectric III	TYPE	\$85.00

C010, Johnson Engineering, 1111 Willow, Oklahoma City, OK, 73159, 5/31/87:

2329485	Remington Rand 101	TYPE	\$85.00
---------	--------------------	------	---------

C008, Red River Realty, 1101 S. Byrd, Purcell, OK, 73346, 5/5/87:

A0064370	Sierra Juki	TYPE	\$65.00
0975687	IBM Selectric II	TYPE	\$65.00

UNIT TEST--Lesson 10

Name: _____

Date: _____

SHORT ANSWER

Complete each of the following by writing the complete dBASE III PLUS command to initiate each of the operations (10 pts. each):

1. Select a database file named INV for work area 1 and a database file named INVTRANS for work area 2 (four commands): _____

2. Set a relation on the common field PART_NO with INVTRANS as the active database: _____
3. Update the target database INV from the source database INVTRANS and replace the ON_HAND field with the ON_HAND in the INV database plus the QTY_REC minus the QTY_SOLD in INVTRANS database:

4. Clear all work areas: _____
5. Delete the database file INV and then rename the database file INVSORT to INV (two commands): _____

UNIT TEST--Lesson 10 Continued

TRUE OR FALSE

Designate each of the following statements as TRUE or FALSE by writing T for True or F for False in the blank to the left of the number (5 pts. each):

- _____ 6. Separate buffers or work areas are provided for each database file. These areas are identified as 0 through 9 or A through J.
- _____ 7. When dBASE III PLUS is first started, the work area is 1.
- _____ 8. To work with multiple database files, a separate work area must be identified as each database file is used.
- _____ 9. Once a file is opened, it may be selected by either the work area number or letter or by use of the file alias or name.
- _____ 10. Two files may be linked on any field in the database files.
- _____ 11. To include a field that is not in the active database in an expression, it must be identified as field->alias.
- _____ 12. When using the UPDATE command, the database being updated must be the active database file in the currently selected work area.
- _____ 13. When using the UPDATE command, the database being updated from must be the active database file in the currently selected work area.
- _____ 14. When using the UPDATE command, the key field does not have to have the same name in both database files.
- _____ 15. To clear the work area, each database file must be closed separately.

UNIT TEST--Lesson 11

Name: _____

Date: _____

SHORT ANSWER

Complete each of the following by writing the complete dBASE III PLUS command to initiate each of the operations (10 pts. each):

1. Output the active database to a text file named, INVTXT, in SDF format using the fields, PART_NO, DESC, and ON_HAND: _____

2. Output the active database to a text file named, INVTXT, using all fields delimited with quotation marks: _____

3. Export the active database to a PFS file named, INVPPS: _____

4. Input the text file, INVTXT, to the active database using the SDF format: _____

5. Import a PFS file named, INVPPS, to the active database: _____

UNIT TEST--Lesson 11 Continued

TRUE OR FALSE

Designate each of the following statements as TRUE or FALSE by writing T for True or F for False in the blank to the left of the number (10 pts. each):

- _____ 6. dBASE III PLUS outputs text files in two formats (not including PFS format).
- _____ 7. All fields will be copied to the target text file using the COPY TO command unless otherwise specified by the FIELDS option.
- _____ 8. Generally, mailmerge programs use files in which the records are stored in SDF format.
- _____ 9. When inputting a text file to dBASE III PLUS, a database structure is created automatically.
- _____ 10. The COPY TO and APPEND FROM commands may not be performed from the Assistant program.

UNIT TEST--Lesson 12

Name: _____

Date: _____

MATCHING

Match the following statements with the appropriate dBASE III PLUS programming command by writing the letter of the command in the blank to the left of the number (5 pts. each):

- | | |
|---|---------------|
| _____ 1. Clears the screen and homes the cursor | a. * |
| _____ 2. Prompts the user for keyboard entry until a carriage return is entered | b. @.SAY |
| _____ 3. Used for comment lines--the line is not executed by dBASE III PLUS | c. CLEAR |
| _____ 4. Creates a <u>loop</u> control structure | d. CLOSE |
| _____ 5. De-activates all active database files | DATABASES |
| _____ 6. Returns control to the program that called it or to the dot prompt | e. DO WHILE |
| _____ 7. Same as saying print or space one line | f. EJECT |
| _____ 8. Print or display information in a specified format | g. IF/ENDIF |
| _____ 9. All succeeding @ commands will be routed to the specified device | h. INPUT |
| _____ 10. Causes the printer to form feed | i. RELEASE |
| _____ 11. Deletes memory variables | j. RETURN |
| _____ 12. Invokes the dBASE word processor | k. SET DEVICE |
| _____ 13. Creates a <u>condition</u> control structure | l. SET TALK |
| _____ 14. Controls screen messages about dBASE processing conditions | m. SKIP |
| _____ 15. Moves the record pointer to the next record | n. ? |
| | o. MODIFY |
| | COMMAND |
| | p. SET |
| | RELATION |

UNIT TEST--Lesson 12 Continued

TRUE OR FALSE

Designate each of the following statements as TRUE or FALSE by writing T for True or F for False in the blank to the left of the number (5 pts. each):

- _____ 16. A command file is another name for a dBASE III PLUS program.
- _____ 17. If no extension is used, dBASE III PLUS automatically adds the extension, .COM, to command files.
- _____ 18. dBASE III PLUS has no built-in function for entering command files.
- _____ 19. dBASE III PLUS **requires** all keywords to be in upper case and all other words in lower case in command files.
- _____ 20. When the program called by a DO is complete, control will be returned to the calling program or to the keyboard if issued from the dot prompt.

UNIT TEST--Lesson 12

Programming with dBASE III PLUS Reference Manual (Optional)

Name: _____ Date: _____

MULTIPLE CHOICE

Complete each of the following statements by writing the letter of the correct answer for each in the blank to the right of the number (10 pts. each):

- _____ 1. Which of the following is not one of the four most common programming mistakes:
A. Misspelling commands
B. Syntax errors, or using a command incorrectly
C. Not capitalizing dBASE commands
D. Issuing an incomplete command

- _____ 2. Which of the following is not a dBASE program control mechanism?
A. Branching
B. Looping
C. Conditions
D. Input/Output

- _____ 3. The dBASE programming concept that there is only one way to get to a submodule and only one exit from a submodule is called:
A. First come, first served
B. One entry, one exit
C. Single exit, single return
D. Singular advancement

- _____ 4. The dBASE programming concept that says you start with the general and gradually work down to the specifics is called:
A. Top-down program design
B. General-to-specific program design
C. Multi-advancement program design
D. Procedural program design

- _____ 5. Which of the following is not one of the four dBASE program control structures:
A. DO WHILE..ENDDO
B. IF..NEXT
C. FOR..NEXT
D. DO CASE

UNIT TEST--Lesson 12 Continued

Programming with dBASE III PLUS Reference Manual (Optional)

- _____ 6. Which of the following is not a type of memory variable:
A. Logical
B. Memo
C. Date
D. Numeric
- _____ 7. Which of the following is not a fullscreen input command:
A. APPEND
B. EDIT
C. BROWSE
D. GET/READ
- _____ 8. Which of the following is not a template symbol used in formatting screen display:
A. A
B. 9
C. !
D. Z
- _____ 9. Which command will send the contents of @..SAY lines to the printer:
A. SET DEVICE TO PRINT
B. SET PRINT ON
C. SAY PRINTER ON
D. SET SAY DEVICE
- _____ 10. Which of the following commands is not a debugging command:
A. SET DEBUG
B. SET HISTORY
C. SET TRACE
D. SET ECHO

UNIT TEST--Lesson 13

Name: _____

Date: _____

TRUE OR FALSE

Designate each of the following statements as TRUE or FALSE by writing T for True or F for False in the blank to the left of the number (5 pts. each):

- _____ 1. Structured design involves using a logical structure for the system and programs.
- _____ 2. Top-down design means that large-scale functions are defined first and then gradually working toward smaller components.
- _____ 3. A feature of structured design is that the functions of the system are organized into distinct modules.
- _____ 4. Each module in structured design may have several entry and several exit points.
- _____ 5. Each module in structured design should be small or limited in size.
- _____ 6. System design is illustrated by an Input/Output Chart.
- _____ 7. In top-down design a higher level module may flow to a lower level module and a lower level module always returns to the higher level module that called it.
- _____ 8. Comments and documentation should not be used in programs because they make the program exceptionally long.
- _____ 9. The DO CASE/ENDCASE structure is like a series of IF/ENDIF structures.
- _____ 10. When writing a dBASE program, it is suggested that all dBASE reserved words are lower case and all other words should be capitalized.
- _____ 11. Do not use dBASE III PLUS reserved words as variable names.
- _____ 12. Reserve the twenty-third line of the screen for error messages.
- _____ 13. Keep user help messages to a minimum as they greatly slow processing.

UNIT TEST--Lesson 13 Continued

- _____ 14. Use indentation in a program to set off control structures.
- _____ 15. Use menus to control the access and operation of programs.
- _____ 16. A valuable programming technique for a menu program is to set up an endless loop using DO WHILE .T.
- _____ 17. The @...GET command must be used with a READ command.
- _____ 18. If a multiple branch is needed in a program such as a menu program, use IF/ENDIF commands.
- _____ 19. Programs should be written before design and analysis occurs.
- _____ 20. Always close database files that are not being used.

UNIT TEST--Lesson 14

Name: _____

Date: _____

TRUE OR FALSE

Designate each of the following statements as TRUE or FALSE by writing T for True or F for False in the blank to the left of the number (10 pts. each):

- _____ 1. To start the Applications Generator, type the command:

.DO APPLICATION GENERATOR
- _____ 2. The Applications Generator is a program written in the dBASE programming language to create other dBASE programs.
- _____ 3. The Applications Generator can create even sophisticated dBASE programs without any knowledge of the dBASE programming language.
- _____ 4. You cannot create database files, screen formats, report formats, or label formats from the Applications Generator.
- _____ 5. If an index file is already set up for a database, the Applications Generator uses it; otherwise, it creates an index file for you.
- _____ 6. The Advanced Applications Generator will allow you to generate custom applications that issue standard dBASE III PLUS commands.
- _____ 7. An application generated by the Applications Generator is mainly a menu program that supervises access to dBASE III PLUS commands or other dBASE programs.
- _____ 8. EXECUTABLE dBASE COMMAND entries on the Advanced Applications Generator screen may be as long as 128 characters.
- _____ 9. You cannot execute an application from the Applications Generator program.
- _____ 10. If you created an application called, DEALERS, the file name listed on the directory would be DEALERS.PRG.

ANSWER KEYS FOR UNIT TESTS (10-12)

LESSON 10-LINKING AND UPDATING MULTIPLE FILES

<u>Short Answer</u>	<u>True or False</u>
1. SELECT 1	6. False
USE INV	7. True
SELECT 2	8. True
USE INVTRANS	9. True
2. SET RELATION TO PART_NO INTO INVTRANS	10. False
3. UPDATE ON ON_HAND FROM INVTRANS REPLACE	11. False
ON HAND WITH ON_HAND+INVTRANS->QTY_REC-	12. True
INVTRANS->QTY_SOLD	13. False
4. CLOSE DATABASES	14. False
5. DELETE INV.DBF	15. False
RENAME INVSORT.DBF TO INV.DBF	

LESSON 11-INPUTTING AND OUTPUTTING TO OTHER PROGRAMS

<u>Short Answer</u>	<u>True or False</u>
1. COPY FIELDS PART_NO,DESC,ON_HAND TO	6. True
INVTXT SDF	7. True
2. COPY TO INVTXT DELIMITED WITH "	8. False
3. EXPORT TO INVPFS TYPE PFS	9. False
4. APPEND FROM INVTXT SDF	10. True
5. IMPORT FROM INVPFS	

LESSON 12-WRITING SIMPLE PROGRAMS

<u>Matching</u>	<u>True or False</u>
1. c	16. True
2. h	17. False
3. a	18. False
4. e	19. False
5. d	20. True
6. j	
7. n	
8. b	
9. k	
10. f	
11. i	
12. o	
13. g	
14. l	
15. m	

ANSWER KEYS FOR UNIT TESTS (12-14)

LESSON 12-WRITING SIMPLE PROGRAMS (Reference Manual)

Multiple Choice

1. c
2. d
3. b
4. a
5. c
6. b
7. d
8. d
9. a
10. c

LESSON 13-SYSTEM DESIGN AND PROGRAM DEVELOPMENT

True or False

- | | |
|-----------|-----------|
| 1. True | 11. True |
| 2. True | 12. True |
| 3. True | 13. False |
| 4. False | 14. True |
| 5. True | 15. True |
| 6. False | 16. True |
| 7. True | 17. True |
| 8. False | 18. False |
| 9. True | 19. False |
| 10. False | 20. True |

LESSON 14-APPLICATIONS GENERATOR

True or False

1. False
2. True
3. False
4. False
5. True
6. True
7. True
8. False
9. False
10. True

STRUCTURE CHART FORM

ANALYST'S NAME: _____ DATE: _____

SYSTEM PROGRAM NAME: _____

DESCRIPTION: _____

FILE STRUCTURE PLANNING CHART

PLANNER'S NAME: _____ DATE: _____

DATABASE NAME: _____ DESCRIPTION: _____

INDEX FILES: _____ REPORT FILES: _____

FIELD	FIELD NAME	TYPE	WIDTH	DECIMALS	PURPOSE/FUNCTION
1					
2					
3					
4					
5					
6					
7					
8					
9					
10					
11					
12					
13					
14					
15					
16					